FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Cooperative navigation of multiple Marine Robotic Vehicles

Nuno Alberto Moreira Maia

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: António Pedro Aguiar

July 31, 2013

Resumo

Com a constante evolução da navegação e exploração, a humanidade sempre quis alcançar mais e elevar o conhecimento humano acerca do mundo para além dos seus limites. Tendo explorado cada continente, muito há ainda por descobrir no fundo dos oceanos. Devido a custos monetários e a ser pouco viável no que diz respeito a dispendiosos recursos humanos, a exploração de ambientes subaquáticos é feita através da utilização de veículos robóticos autónomos. Esta possibilidade não só é mais segura, não colocando em risco vidas humanas em territórios inóspitos, mas também é, em alguns aspectos, mais fácil de implementar. Como estes veículos são não tripulados, eles precisam de ser autónomos e vigiados à distância. Isto levou à importância do desenvolvimento de técnicas de navegação.

Em terra, esta navegação é possível através do sistema de posicionamento já bem conhecido, GPS. No entanto, debaixo de água é impossível usar este sistema devido ao facto de esses sinais não se propagarem bem neste meio. Há a necessidade de utilizar outro tipo de sinais, que neste momento são ondas acústicas pois se propagam bem nesse meio. Para calcular a localização do veículo subaquático, a ideia principal é usar o sistema de emissores/receptores que enviam e recebem do veículo o sinal acústico. Através da medida do tempo de viagem e sabendo a velocidade do sinal acústico, é então possível encontrar a localização do objecto. Em 2D (ou seja, sabendo a profundidade) só são necessários três emissores/receptores (assumindo certas condições espaciais) para descobrir a localização: um dá uma circunferência de possíveis localizações, dois dão duas circunferências interceptadas em dois pontos, o que significa duas localizações possíveis, e o terceiro intercepta uma destas localizações, que é a localização correta do veículo, este processo é designado trilateração. No entanto, o vasto número de emissores/receptores que são usados é dispendioso, sentindo-se a necessidade de encontrar uma maneira de calcular a localização de veículos utilizando menos emissores/receptores.

Com a utilização de apenas um emissor/receptor e implementando um algoritmo que calcula o deslocamento do veículo em diferentes tempos, é possível criar uma rede virtual de emissores/receptores que vai ajudar a encontrar uma estimativa da posição do veículo. Esta estimativa da localização é mais tarde filtrada usando o filtro de Kalman. O papel deste filtro é, não só filtrar o ruido, mas também complementar o resultado da trilateraçao com informação da velocidade linear relativa obtida por um doppler e para lidar com possíveis perturbações desconhecidas tais como as correntes. De facto, testes foram feitos e mostraram que sem a utilização do filtro de Kalman e sob a influência destas correntes, o algoritmo utilizado diverge, não dando os resultados pretendidos.

Depois de ter o algoritmo de navegação implementado para um veículo, mais uma camada de complexidade foi adicionada ao incluir outro veículo com o papel de emissor/receptor (emissor/receptor com movimento) no sentido de tornar a navegação cooperativa. O veículo que funciona como emissor/receptor para outros veículos pode ser um veículo robótico de superfície e por isso, deste modo, ser capaz de utilizar o sistema GPS. Pode também ser um veículo submarino robótico autónomo que serve de emissor/receptor a outros e dos quais a sua posição também pode ser calculada usando uma lógica semelhante à utilizada na navegação singular.

Abstract

With the increased evolution in navigation and exploration, mankind always aimed to go further and push the human knowledge of the world to its limits. Having explored every continent and land surface, much is still left to be discovered underneath the oceans. Due to monetary costs and not being viable due to being too costly in human resources, the exploration of underwater territories is being carried out through the use of autonomous robotic vehicles. This possibility not only is safer, not endangering human lives in inhospitable locations, but it also is in some aspects easier to implement. As these vehicles are unmanned, they need to be autonomous and surveyed from a distance. This led to the importance of the development of navigation techniques.

On land, this navigation is possible through the well-known GPS system. However, underwater it is impossible to use this system due to the fact that those signals do not propagate well in such an environment. There is the need to use another type of signals, which presently turns out to be acoustic waves since these signals propagate well in this environment.

To compute the location of an underwater vehicle, the main idea is to use a system of transponders/beacons that send to and receive from the vehicle an acoustic signal. By measuring the time of travel and knowing the speed of the underwater acoustic signal (AS) it is then possible to find the location of the object. In 2D, (which means knowing the depth) it only takes three beacons (under some spatial assumptions) to find out the location: one gives a circumference of possible locations, two give 2 circumferences which are intercepted in two places, which mean two possible locations and the third one intercepts one of these locations, which is the right location of the vehicle, this process is called trilateration. However, the vast amount of transponders (beacons) that are used is very costly, so there is a need to find a way to compute the location of the vehicles using fewer resources.

With the use of only one beacon, and implementing an algorithm that computes the displacement of the vehicle in different times, it is possible to create a virtual net of beacons which will help finding out an estimate of the vehicle location.

This estimate of the location is later filtered using a Kalman Filter (KF). The role of this filter is to not only filter out the noise, but it is also to complement the result of the trilateration with the relative linear velocity information provided by a doppler and to deal with possible unknown underwater current disturbances. In fact, tests were done that showed that without using the KF and under the influence of currents, the algorithm would diverge, not producing the results intended.

After having the Single Beacon Navigation (SBN) for one vehicle implemented, one more layer of complexity was added by including another vehicle with the role of the beacon (moving beacon) in order to make the navigation cooperative. The vehicle that works as a beacon to the other vehicles can be an Autonomous Surface Craft (ASC) which floats at the surface of the sea and therefore uses the GPS system. It can also be an Autonomous Underwater Vehicle (AUV) which serves as a beacon to others and whose position is computed using a similar logic as the one in the SBN algorithm.

Acknowledgements

An enormous thank you to my supervisor António Pedro Aguiar, for his patience, support, guidance, and availability, despite always having a full schedule. It was important to have a good supervisor, since I knew little about this topic and programming was never my best skill (which is why this dissertation helped me so much in improving my skills and knowledge as I knew close to nothing about what to do).

A big cheers to my father, who was my -'at home' supervisor, helping me understand some important aspects of what I had to do and helping me think when I couldn't find the right way alone.

I would like to thank my family for their help, not only during this semester, while I was doing the dissertation, but also for supporting my choices, even if it meant going to Romania and Brazil for six months each.

Thanks to Agostinho Rocha, André Reis and José Oliveira, for their help in some aspects I needed to learn about in order to complete this dissertation.

And finally, thanks for these great years in college, Tiago Maia, Fabio Silva, Alexandre Santos, Miguel Ribeiro, Luciano Santos, Nuno Guimarães, Bruno Pereira, and the list goes on; without a doubt these years will be missed and remembered with the willingness to go back and relive it all.

Nuno Alberto Moreira Maia

"Life begins at the end of your comfort zone."

Neale Donald Walsch

Contents

List of Figures

List of Tables

Nomenclature

AS	Acoustic Signal
ASC	Autonomous Surface Craft
ASV	Autonomous Surface Vehicles
AUV	Autonomous Underwater Vehicle
CN	Cooperative Navigation
COTS	Commercial off-the-shelf
DE	Doppler Effect
FEUP	Faculdade de Engenharia da Universidade do Porto
GDM	Gradient Descent Method
GPS	Global Positioning System
ISR	Robotic Systems Institute
KF	Kalman Filter
LAUV	Light Autonomous Underwater Vehicle
LBL	Long Baseline
LSTS	Underwater Systems and Technology Laboratory
MARES	Modular Autonomous Robot for Environment Sampling
MER	Mars Exploration Rover
NIBs	Navigation and Instrumental Buoys
NM	Newtow's Method
NP	Noptilus Project
OM	Optimization Method
RV	Robotic Vehicles
SB	Single Beacon
SBN	Single Beacon Navigation
SN	Single Navigation
VNB	Virtual Net Beacon
VB	Virtual Beacon

Symbols

ω	Angular Velocity
s	Seconds
v	Linear Velocity

Chapter 1

Introduction

With the increasing number of unmanned vehicles used in hardly accessible locations in missions, reconnaissance or other exploration objectives, it began to be felt a need for cooperation between those vehicles, and in particular, the use (directly or implicitly) of the navigation sensors from the neighbour vehicles to obtain better navigation performance. The concept of navigation is the act of computing for a vehicle the evolution of its linear and rotational positions and velocities. In the underwater medium, this is a challenging task because GPS is not available. To counter this problem we will use Acoustic Signals (ASs), combined with the Doppler Effect and measurements provided by inertial measurement unit sensors to be able to track the underwater location of the vehicle. After the navigation problem is sorted out we can start implementing cooperative behaviour (Figure 1.1) in the vehicles, such as location feedback, formation abilities and many other possibilities are unlocked.

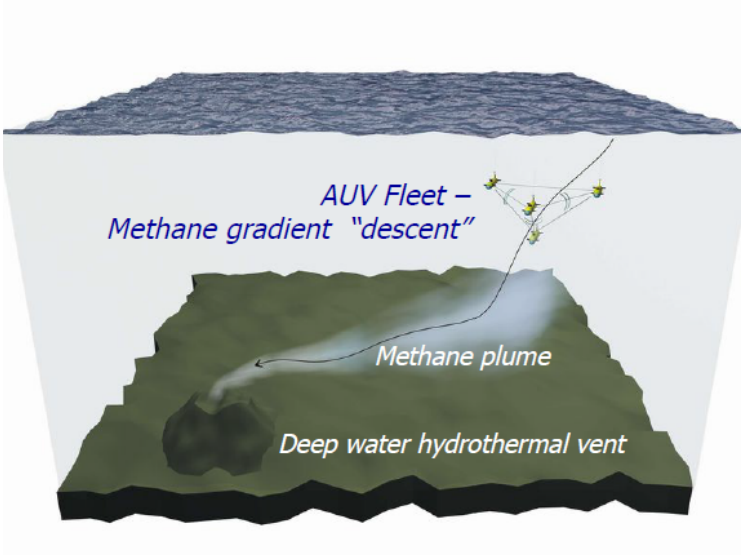

Figure 1.1: AUV fleet in underwater exploration (Source: [1])

1.1 Motivation

Underwater navigation is a critical and difficult task (Fig. 1.2). In this thesis we propose a navigation algorithm for AUVs that relies on single beacon acoustic navigation techniques and takes into account the presence of unknown ocean currents. This idea has advantages compared to the usual method implemented that uses three beacons since relying on multiple beacons means the need to survey more locations, and the deploy, survey and recovery of these beacons,which is costly and time expensive. After implementing the SNB approach it is then possible to make the navigation cooperative which enables a set of different abilities for the group of vehicles.

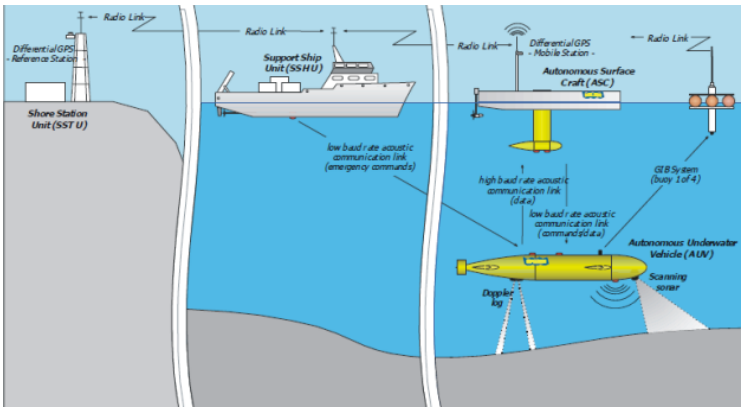

Figure 1.2: Communication between AUV, ASC and station unit (Source: $http://goo.gl/Qrb2t$)

1.2 Objectives

The main objective of this work was to to study, develop and implement cooperative navigation algorithms for marine vehicles. In particular, the following tasks were addressed:

- Literature review and study of estimation algorithms for navigation.
- Study and implementation in Matlab of a single beacon acoustic navigation algorithm.
- Study and design of cooperative estimation strategies for navigation of multiple autonomous Robotic Vehicles (RVs).
- Implementation and evaluation of the algorithms developed through computer simulations.

1.3 Main contribution

The main contributions of this thesis are twofold:

- Implementation and performance evaluation through computer simulations of a single beacon navigation algorithm for single AUVs based on the work of [2]. The algorithm is composed by two main components: the construction of net of virtual beacons and the computation of a first rough position through trilateration, and second a fine estimation of the position using a Kalman filter.
- The extension of the single navigation system to cooperative navigation where in this case we have at least one vehicle playing the role of a moving beacon (contrary to the first case where the beacon is stationary) serving other vehicles for navigation that interact with it.

1.4 Thesis Outline

The remainder of this dissertation is organized as follows:

- Chapter 2 describes the State of the Art starting with a brief historical explanation as well as the present technologies of this topic.
- Chapter 3 presents the implementation of the SBN.
- Chapter 4 describes the extension of the SB algorithm to multiple vehicles using a cooperative strategy. Two cases are studied in detail: CN with AUV/ASC and CN with two AUVs.
- Chapter 5 presents the results and the critical analysis of the algorithm simulations.
- Chapter 6 concludes this thesis with a summary of the results obtained and benefits and drawbacks of the methods proposed, and suggests directions for future investigation.

Chapter 2

State of the Art

2.1 Some well-known historic examples

Throughout the history of exploration and navigation, there are some marks that are worth mentioning. Without a doubt, exploration has been in the eager mind of mankind all along. At first, even though the navigation and the positioning systems of the time were not very precise or sophisticated, the need to travel beyond the unknown pushed us through the oceans and, continent after continent, zone after zone, the knowledge of the entire planet started adding up and fitting in like pieces of a puzzle. We can say that this helped greatly in technological progress, not only due to the instruments that were invented to make travelling possible and finding the way people wanted to travel; but also by making the world more united and enabling the findings of each country, zone, place available everywhere.

2.1.1 Space

The most recent and remarkable event of exploration and navigation is the Mars Exploration Rover Mission (MER) conducted by NASA, which consisted in sending two rovers, Spirit (MER-A) and Opportunity (MER-B) to explore Mars' surface and geology. See Figure 2.1. Basically, the goal of this mission is research related to the past existence of water activity (always mankind's number one goal because it is intimately related to life and the possibility of Mars being a future place for people to live in). This project is a very costly investment in exploration, summing up to several hundreds of million dollars.

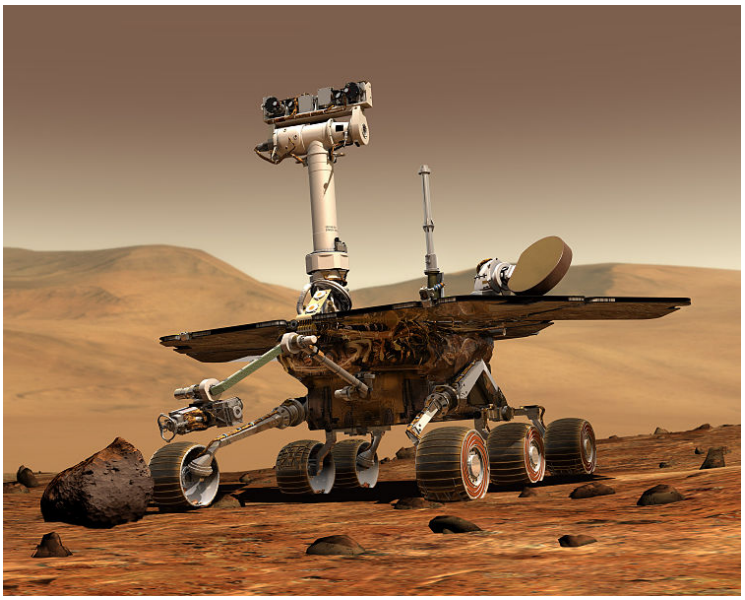

Figure 2.1: MER exploring Mars surface (Source: $http://goo.gl/ILOzi$)

2.1.2 Land

Robots are used to explore places that are dangerous to man. An example of this are the robots used to explore volcanoes and big, deep caves that are at risk of collapsing. Many projects are currently under development for monitoring volcanic activity, which would otherwise be close to impossible. See Figure 2.2. They are used to make crucial observations and taking measures which can determine the future activity of the volcanoes, knowledge that can therefore save more lives in the future. These kinds of robots have got a wide set of tools which help in making eruption scouting possible, such as ways to collect samples, sensors for data acquisition and such. The very first robotic prototypes were used in this field, and after scientists saw their success and potential, the idea of exploring other planets in a similar way, with the improvement of technology, came to mind. [3]

Another more recent example used in volcano exploration is the Robovolc, a versatile robot with more mobility and independence than its counterpart, with a better set of tools and which does not rely as much on other devices and works wireless. [4]

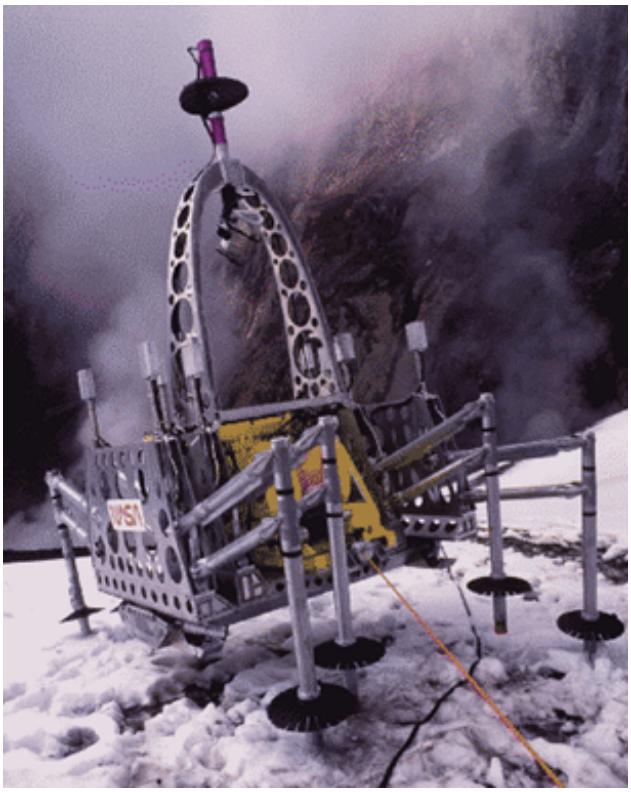

Figure 2.2: One of the first robotic volcanoes explorers (Source: $http://goo.gl/8Ts6A$)

2.1.3 Air

There is also a very important use for air vehicles, in robotic navigation. For instance, there are some tasks that regularly require the use of unmanned aerial vehicles. For example, they are used for ground surveillance, space reconnaissance, to find people in rescue operations and to monitor traffic. See Figure 2.3. For such operations it is common to use not single unmanned vehicles but fleets, which do CN with one another. This kind of navigation enables formation flight that is very useful for scouting wider areas and allows air-refuelling. [5]

Figure 2.3: Aerial robot explorer (Source: $http://goo.gl/8efBh$)

2.1.4 Ocean

The ocean is still an environment that is very unknown to man. Covering the vast majority of the Earth, only a very small portion of it has already been scouted. Due to their inhospitable nature, it was not possible to explore deep seas up until recently. Shallow locations can be explored by humans, but the majority of the ocean is very deep and cannot be explored by divers. Even manned vehicles are inferior to RVs when it comes to ocean exploration. In this case, robotic exploration became a quick reality. Considering the fact that the deepest oceans potentially keep some of life's most important secrets, this field is bound to be very popular in the future of exploration. See Figure 2.4. Some vehicles called gliders are used to collect data from the oceans being able to cover distances up to 600 km and representing a large contribution to future research concerning these environments. See Figure 2.5. [6] [7]

Figure 2.4: Underwater robotic explorer (Source: $http://goo.gl/CHXrl$)

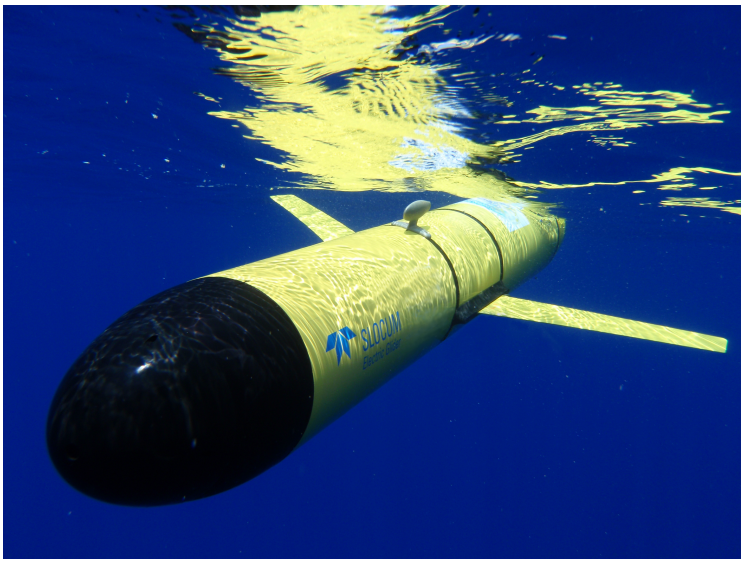

Figure 2.5: Glider (Source: $http://goo.gl/VlSfa$)

2.2 Doppler Effect

As any object moves through the fluid, the fluid near the object is disturbed. The disturbances are transmitted through the fluid at a distinct speed called the speed of sound. See Figure 2.6.

As explained by NASA, sound moves through the fluid as a series of waves. The distance between any two waves is called the wavelength and the time interval between waves passing is called the period. The wavelength and the frequency are related by the speed of sound; high frequency implies short wavelength and low frequency implies a long wavelength. Shorter wavelengths produce higher pitches. In an ideal fluid, the speed of transmission of the sound remains a constant regardless of the frequency or the wavelength. The speed of sound only depends on the state of the fluid (or gas) not on the characteristics of the generating source. [8]

They say that, because the speed of sound depends only on the state of the gas, some interesting physical phenomena occurs when a sound source moves through a uniform gas. As the source moves it continues to generate sound waves which move at the speed of sound. Since the source is moving slower than the speed of sound, the waves move out away from the source. Upstream (in the direction of the motion), the waves bunch up and the wavelength decreases. Downstream, the waves spread out and the wavelength increases. The sound that our ear detects will change in pitch as the object passes. This change in pitch is called a Doppler Effect (DE). There are equations that describe the DE. For example in the air, as the moving source approaches our ear, the wavelength is shorter, the frequency is higher and we hear a higher pitch. If we call the approaching frequency f_a, the speed of sound a, the velocity of the approaching source u, and the frequency of the sound at the source f, then

$$f_a = \frac{fa}{a-u} \tag{2.1}$$

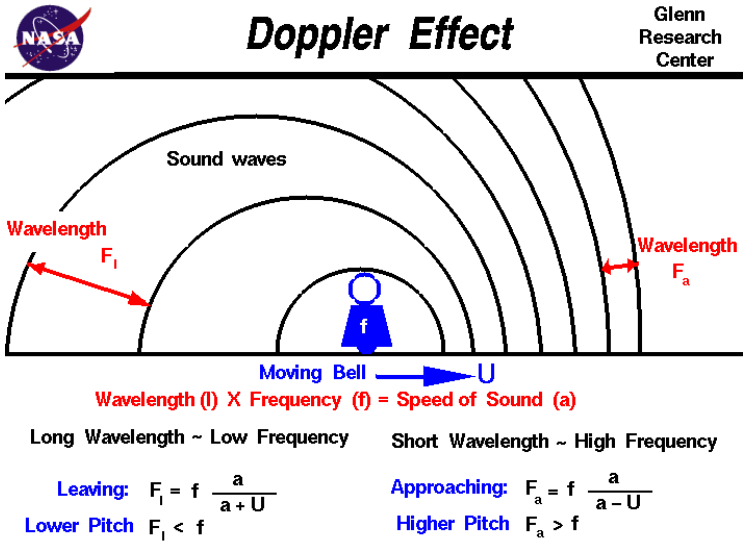

Figure 2.6: Doppler explanation (Source: $http://goo.gl/k9fzz$)

Putting it simply: the DE is the change in frequency of a wave, from an observer moving relative to its source. This happens as the source of the waves is moving toward the observer. For instance, every wave crest is emitted from a closer range, arriving quicker, therefore reducing the time between waves and increasing its frequency. See Figure 2.7.

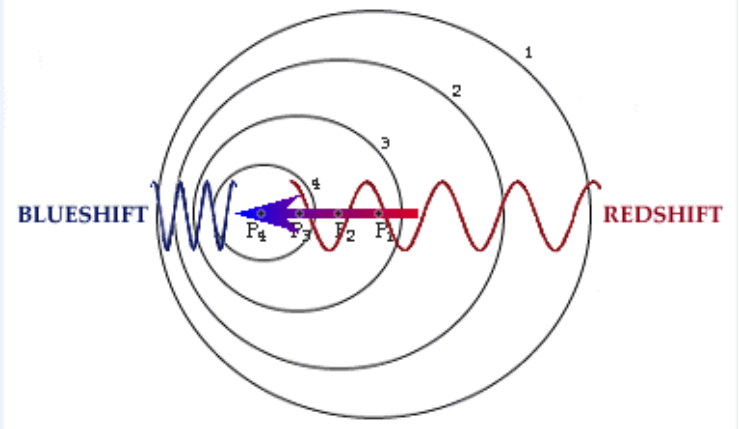

Figure 2.7: Doppler functioning (Source: $http://goo.gl/627KC$)

Depending on the medium in which they propagate, the velocity of the source and observer varies. [9]

The DE can result from

- Motion of the source;
- Motion of the observer;
- Motion of the medium.

In our case, underwater vehicles, the Doppler Shift of a target is used to determine the speed of a submarine using both passive and active sonar systems. See Figure 2.8. As a submarine passes by a passive sonobuoy, the stable frequencies undergo a Doppler shift, and the speed and range from the sonobuoy can be calculated. If the sonar system is mounted on a moving ship or another submarine, then the relative velocity can be calculated.

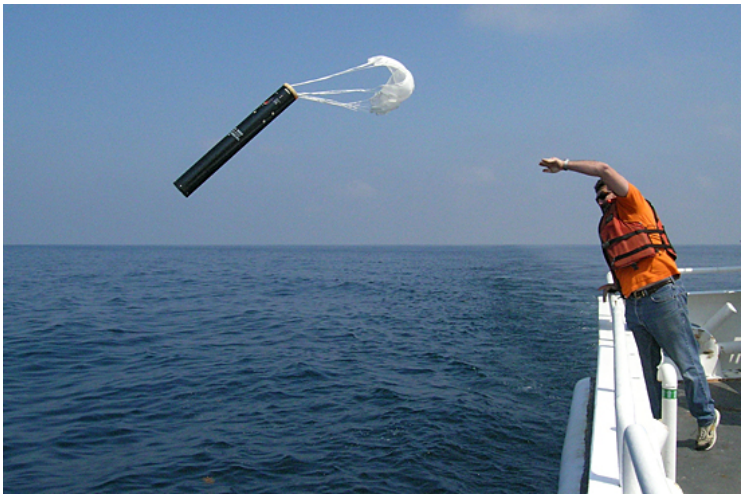

Figure 2.8: Dropping Sonobuoy (Source: $http://goo.gl/uI2Nm$)

2.3 Current Underwater Navigation Systems

Nowadays, underwater navigation is a complex concept that involves many variables.

As explained, acoustic waves, resulting from vibration in water particles, are used for navigation and communication between vehicles.

AUVs are equipped with an omni-directional transducer which is capable of sending and receiving acoustic signals, managing to locate its position. The AUV sends signals to beacons located nearby and its location is calculated based on the time taken for the signals to travel, as the speed of the acoustic signal can be easily computed. Some research about the subject was important to understand this system better. [10] [11] [12]

A standard formula to compute the sound speed underwater is:

$$c = 1449.3 + 4.572 * T - 0.0445 * T^2 + 0.0165 * d + 1.398 * (S - 35) \tag{2.2}$$

This computing is not as simple as one would expect since there are many factors to take into consideration. [13]

This equation remains valid for this set of values:

- Temperature T = -3 to 30 ºC;
- Depth d = 0 to 10,000 meters;
- Salinity S = 33 to 37 parts per thousand.

At the surface the precision of this equation is inferior to 0.2 percent. Usually the sound speed used is the one for T = 13 ºC of about 1500 m/s. [14]

2.3.1 Long Baseline

LBL is a system used for locating AUVs, which usually uses three or four beacons that are strategically placed within the operation area. See Figure 2.10. The AUV, in order to get its location, sends a different acoustic signal to each beacon and receives the response signal, computing the distance to each of these beacons based on the sound speed underwater (mentioned above). See Figure 2.9. With the distance to each of the beacons, it is easy to compute its relative location. Having previously downloaded the actual coordinates of each beacon to the AUV, it is able to acquire its absolute location. [15]

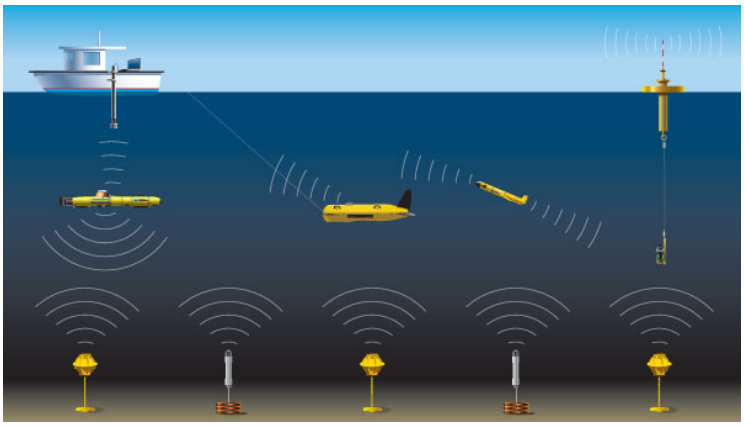

Figure 2.9: LBL (Source: *http://goo.gl/VVBQa*)

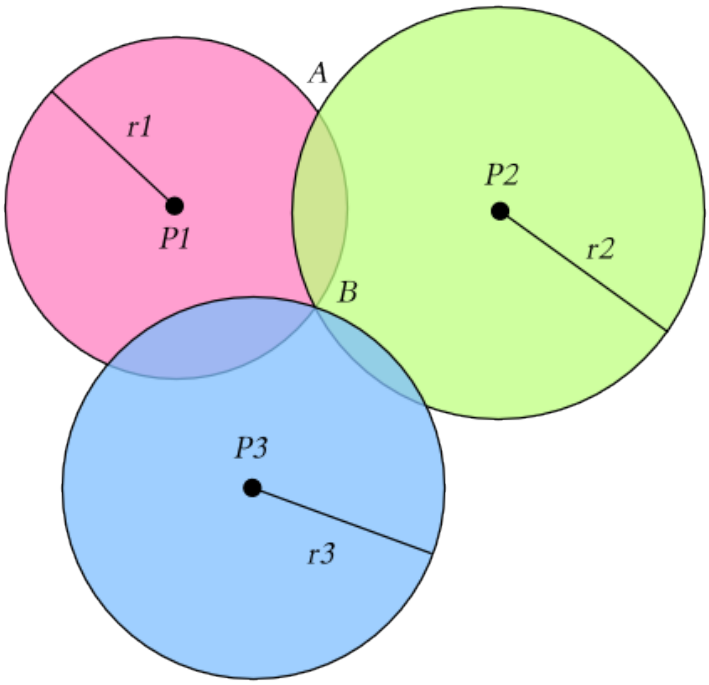

Figure 2.10: Position obtained using 3 beacons trilateration (Source: http://goo.gl/uiXuX)

2.3.2 Marine robotic platforms

2.3.2.1 AUV

There are a great number of AUVs in the world. Two examples of them are the ones that have been developed at FEUP.

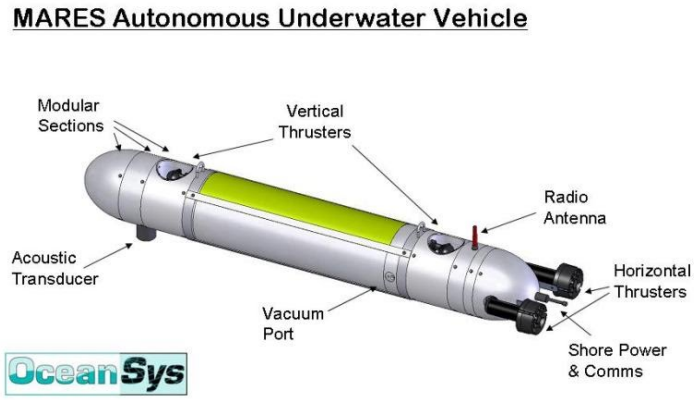

Figure 2.11: FEUP AUV MARES (Source: $http://goo.gl/pxa1B$)

One AUV, called Modular Autonomous Robot for Environment Sampling (MARES), was developed by "The Ocean Systems Group"(Oceansys) of FEUP, together with ISR-Porto (Robotic Systems Institute). See Figure 2.11. [16]

According to OceanSys, MARES (Modular Autonomous Robot for Environment Sampling) is a highly modular autonomous underwater vehicle, designed for shallow water operations. The vehicle can be configured to carry a wide variety of oceanographic sensors and includes a set of navigation sensors to ensure that the predefined trajectories are followed.

The characteristics of MARES are in the table 2.1

Table 2.1: Characteristics of MARES

Characteristics of MARES	
Length	1.5 m
Diameter	20 cm
Weight in Air	32 kg
Depth Rating	100 m
Horizontal Velocity	$0-2$ m/s, *variable*
Autonomy/Range	*about* 10 *hrs* / 40 km

MARES has a very particular feature - the horizontal motion and the vertical motion are completely independent because it is propelled by two vertical thrusters and two horizontal ones. This rare feature makes it possible for the AUV to stay still in the ocean, underwater.

Some features of MARES include:

- On-board location sensors and movement aid;
- Modular construction, with reconfigurable sections;
- Spare ports to accommodate payload sensors;
- Compact and lightweight - fits the trunk of a car;

- Robust, with fully shrouded moving parts;
- Operates in confined areas - can ascend/descend in the vertical;
- Autonomous operation, with simple mission definition;
- Rechargeable Li-Ion batteries.

Another very interesting AUV is the Light Autonomous Underwater Vehicle (LAUV) (Fig. 2.12 developed by the Underwater Systems and Technology Laboratory (LSTS). [17] According to LSTS, the LAUV is an Autonomous Underwater Vehicle targeted at innovative standalone or networked operations for cost-effective oceanographic, hydro-graphic and security and surveillance surveys.

This lightweight vehicle can be easily launched, operated and recovered with a minimal operational setup. The operation of the LAUV does not require extensive operator's training. It is an affordable, highly operational and effective surveying tool. Starting at a basic functional system that includes communications, computational system and basic navigation sensors, the LAUV capabilities are built up adding optional payload modules. [18]

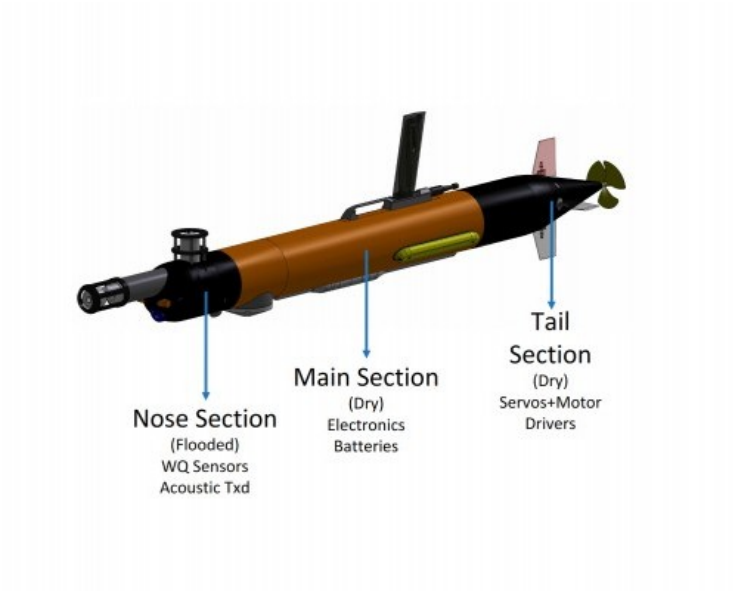

Figure 2.12: Light Autonomous Underwater Vehicle (Source: $http://goo.gl/mrXai$)

Table 2.2: Characteristics of LAUV

Characteristics of LAUV	
Length	*Starting at* 110 *cm* (*depends on configuration*)
Diameter	15 *cm*
Weight	*Starting at* 18 *kg*
Endurance	*Up to* 8 *hours* @ 3 *knots*
WiFi	2.4 *GHz and/or* 5 *GHz*
GSM/HSDPA	*Quad − band* 3*G module*
Maximum Depth	100 *meters*
INS	*MGB* : 1 *degree per hour*
CTD	*Up to* 6*Hz sampling rate*
Echo Sounder	*Frequency* : 675 *kHz* (*Mounting pointing forward*)
Camera	*Resolution* : 720*p in H*.264 *and near* 1080*p in JPEG*
Multibeam	*Frequency* : 260 *kHz*. *Range* : 100 *meters*
SideScan Sonar	*Single or dual frequency*

Some features of the LAUV are:

- Security and Surveillance;
- Oceanography;
- Hydrography;
- Inertial Navigation.

2.3.2.2 ASVs or ASCs

FEUP has also developed small size autonomous surface vehicles (Zarco and Gama), with a catamaran shape, designed to operate in quiet waters (rivers, lakes, dams).As mentioned in OceanSys, in its basic configuration, each vehicle weights around 50 kg, and has an additional payload capacity of 20 kg. See figure 2.13. The vehicle is actuated by two electrical thrusters and can operate at speeds up to 3 knots. It carries an on-board computer to run fully autonomous or remotely controlled missions, while storing all internal and payload data. A WiFi link connects the ASV to a shore station, allowing for real-time data transmission and mission supervision.

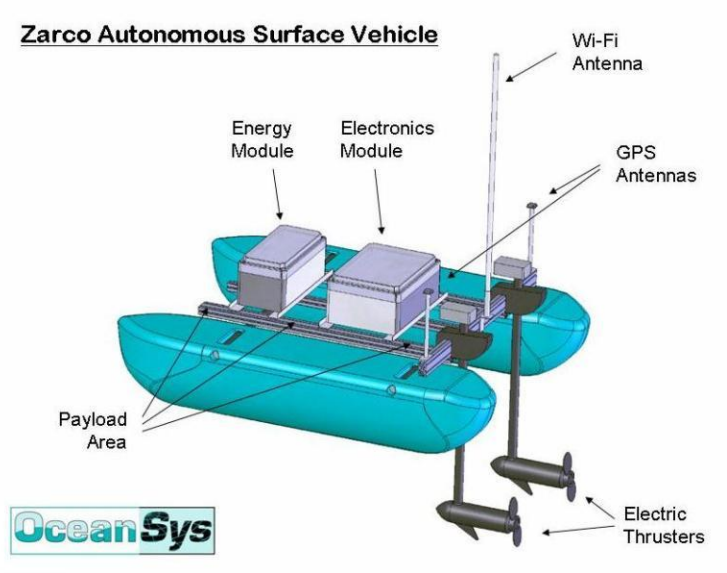

Figure 2.13: ASV Zarco (Source: $http://goo.gl/pxa1B$)

Table 2.3: Characteristics of the ASVs or ASCs

Characteristics of the ASVs or ASCs	
Length	1.5 m
Width	1 m
Weight in Air	50 kg
Net Buoyancy	50 kg
Horizontal Velocity	$0-3$ $knts$, *variable*

Some features of ASCs or AUVs:

- Modular mechanical structure, using COTS elements;
- Separation of major modules without tools;
- Spare ports to accommodate payload sensors;
- Very stable platform;
- Can operate autonomously or remotely controlled;
- Real-time transmission of data to shore.

2.3.2.3 Buoys

The Buoys that are used for navigation and instrumentation are called Navigation and Instrumentation Buoys (NIBs). See figure 2.14

NIBs are moored floating platforms with on-board electronics and energy management system. The basic configuration includes rechargeable Lead-acid batteries, a compact GPS receiver and a

low-power radio modem. NIBs can carry a great variety of sensors and transmit data in real time to a shore station using the radio link.

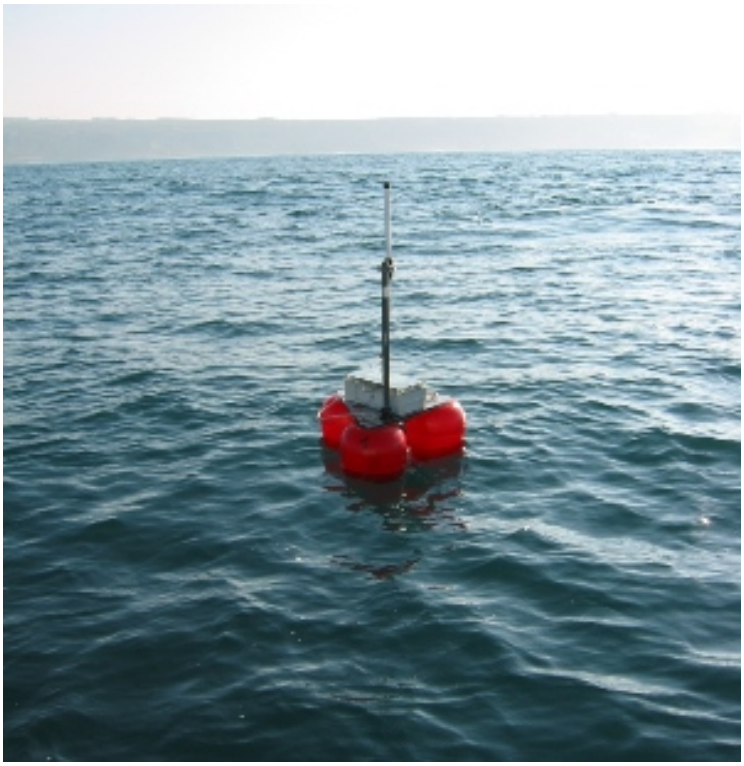

Figure 2.14: Navigation Instrumentation Buoy (Source: $http://goo.gl/pxa1B$)

NIBs, based on OceanSys, are used as acoustic navigation beacons for the AUVs. In this scenario, they have electronic boards to receive and decode acoustic signals sent by the vehicle and respond by transmitting other coded pings into the water. Since they are deployed in known positions, the vehicle can determine its position by triangulation. During an AUV mission, the buoys also relay navigation information back to a mission control station, allowing for vehicle tracking and mission supervision. [16]

Table 2.4: Characteristics of the Buoy

Characteristics of the Buoy	
Overall Diameter	75 *cm*
Weight in Air	30 *kg*
Net Buoyancy	20 *kg*
Antenna Height	1 *m*

2.3.3 Recent Projects

There are many projects currently being worked on related to AUVs in general and evolving towards multi-AUVs systems.

2.3.3.1 Noptilus Project

One good example of this is the Noptilus Project (NP). The evolution of multi-AUV systems is still in progress of maturing and it still has some flaws when faced with real-life complex situation-awareness operations. These types of operations often involve complex reasoning and demand the ability to make decisions and, as such rely on human beings.

As we are well aware of, any system that involves human beings is bound to have flaws related to the very condition of mankind.

NOPTILUS is a project to fight this flaws. It represents an effective completely autonomous multi-AUV concept which is 100 percent autonomous, thus not relying on humans.

Referring to their own words, in order for this to happen, some advances are required on some aspects:

- Cooperative and cognitive-based communications and sonars (low level);
- Gaussian Process-based estimation ;
- Perceptual sensory-motor;
- Learning motion control (medium level):
- Learning/cognitive-based situation understanding and motion strategies (high level).

Of extreme importance is the integration of all these advances and the demonstration of the NOPTILUS system in a realistic environment at the Port of Leixões, utilizing a team of 6 AUV's that will be operating continuously on a 24hours/7days-a-week basis.

Evaluation of the performance of the overall NOPTILUS system will be performed with emphasis on its robustness, dependability, adaptability and flexibility especially when it deals with completely unknown underwater environments and unpredictable situations as well as its ability to provide with close to optimal performance. [19]

Chapter 3

Navigation for single Marine Robotic Vehicles

This chapter describes the single beacon navigation algorithm for single AUVs based on the work of [2]. The algorithm can be split into two main components: the Trilateration and the KF.

3.1 System overview

The system is composed by some principal blocks which constitute the navigation process. See Figure 3.1.

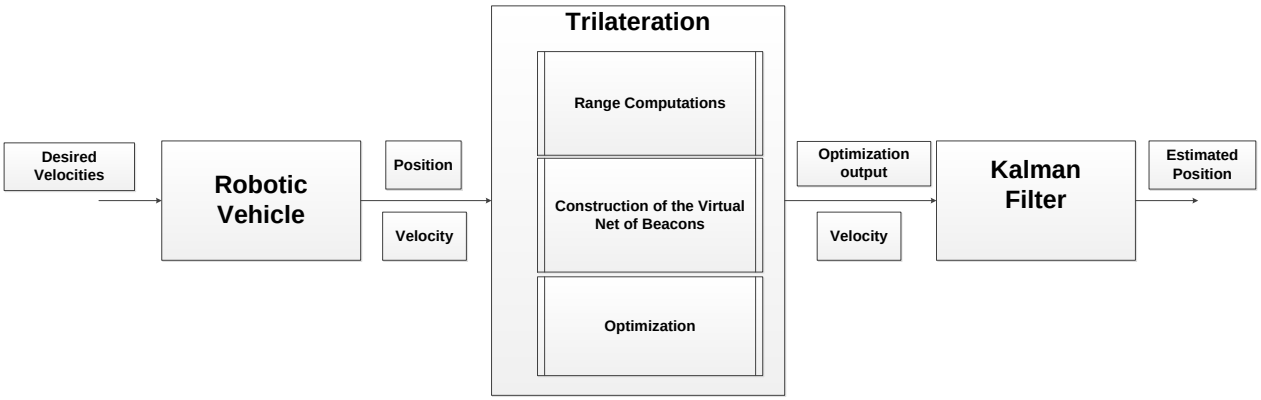

Figure 3.1: System block diagram

3.2 Kinematic vehicle model

To test the single beacon navigation algorithm in Matlab, first a kinematic model of an AUV (in the horizontal plane), was implemented. To this end, a subsystem was defined and implemented as a Matlab block. The inputs of the vehicle block are the linear and angular velocities. The block implements the following equations:

$$\dot{x} = vsin(\psi) + v_{cx} \tag{3.1}$$

$$\dot{y} = vcos(\psi) + v_{cy} \tag{3.2}$$

$$\dot{\psi} = w \tag{3.3}$$

where (x,y) is the vehicle position and ψ its orientation. For simplicity we omitted the third position coordinate z in the formulation. We assume that the AUVs are equipped with pressure sensors that provide the underwater depth. In the equations 3.1, 3.2 and 3.3, we have included explicitly the current $v_c = (v_{cx}, v_{cy})$, which is assumed to be constant but unknown from the navigation algorithm.

3.3 Trilateration

Trilateration is the process of determining absolute or relative locations of points by measurement of distances, using the geometry of circles, spheres or triangles.

This section describes the several components of the trilateration system. Firstly, there is the formation of the Virtual Net of Beacons, using ranges and displacements that occur in the system AUV-beacon. With this set, we then compute the position of the virtual beacons, which will later be used in an optimization process that gives an estimation of the position of the vehicle that we are looking for.

3.3.1 Forming the Virtual Net of Beacons

To obtain a virtual net of beacons, firstly we proceed to compute the spatial movement of the vehicle from each position to the following one. To do this we integrate the values of the relative linear velocity of the vehicle. In Matlab this was done by implementing the simple discrete formula $p_{k+1} = p_k + \delta v_k$ where p is the position of the vehicle, δ is the sampling time (which in our case is $\delta = 0.1s$) and v the linear velocity.[2]

The navigation system used consists in one beacon, as mentioned before. However in the process of determining the various positions of the AUV over time, some virtual beacons (VBs)

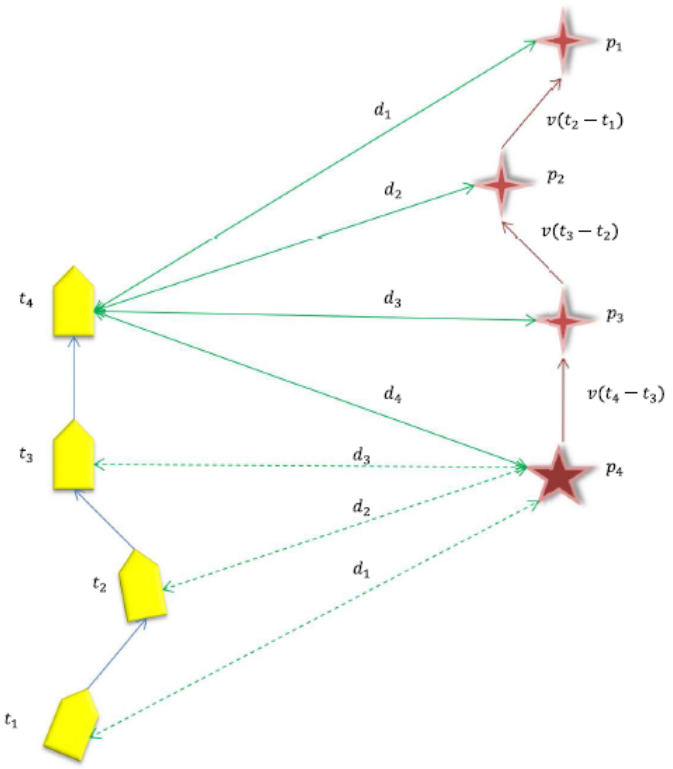

Figure 3.2: Network of beacons creation (Source: [2])

are created and used in order to make the determination of its location possible. In what follows, we consider a virtual net of beacons composed by $N = 4$ beacons.

The process begins in the real beacon (P_4) that creates the VBs using the displacement made by the vehicle. This is done over 4 different times in 3 different directions. As it is clear from Figure 3.2.

The vehicle obtains the range to the beacon every T seconds (which in our case is $T = 4s$). This is done by sending an AS, and knowing its speed and time of travel. During the intervals between the range measurements, the vehicle displacement is also saved by integrating the velocity.

After the 4 distances to the beacon are computed, the VBs are created. Firstly, P_3 is placed, by doing the inverse motion that the AUV did from t_3 to t_4. This motion corresponds to $v(t_4 - t_3)$. Once computed this motion, and summing to the beacon (P_4) we find the position of the first VB (P_3). From this new VB we will follow the same logic applying the motion from t_3 to t_2 (and computing : $v(t_3 - t_2)$), creating the VB P_2 and the same goes for the last interval t_2 to t_1 which creates the VB P_1. [20]

Having the beacons, the next step is to compute, by trilateration, the position of the vehicle.

3.3.1.1 Position of the VBs

Formally, by following the process described above, the positions of the VBs are obtained according to:

$$p_i = p_N + \int_{t_i}^{T_N} v(\tau) d\tau \tag{3.4}$$

Where P_N is the location of the actual transponder, and v the AUV linear velocity and t_i the time when the measurements are taken.

For the particular case of our implementation (N=4) we have

$$p_4 = d(t_4) \tag{3.5}$$

$$p_3 = p_4 + int_{v34} \tag{3.6}$$

$$p_2 = p_3 + int_{v23} \tag{3.7}$$

$$p_1 = p_2 + int_{v12} \tag{3.8}$$

Having computed this, it is time to find the best estimate for the vehicle's position $\bar{p}$. In order to do this, a method of optimization is used.

The expression to optimize is the one that follows and is based on the main reference article for the SB implementation.

$$\bar{p} = \arg\min_{p \in R^2} \sum_{i=1}^{N} (||p_i - p||^2 - d_i^2)^2 \tag{3.9}$$

3.3.2 Optimization

The optimization of a function is the selection of a best element. It consists in maximizing or minimizing the function considered. See Figure 3.3. In our particular case, we are going to optimize the function in (3.9) by finding its minimum. There are several methods used to do this. However, some are more appropriate for some cases than others.

Two typical optimization procedures are

- Gradient Descent Method (GDM);
- Newton Method (NM).

3.3.2.1 Gradient Descent Method

The GDM is a first-order OM. It is used in order to find local minimum and maximum values, taking steps proportional to the negative or positive gradient of the function at the current point, accordingly.

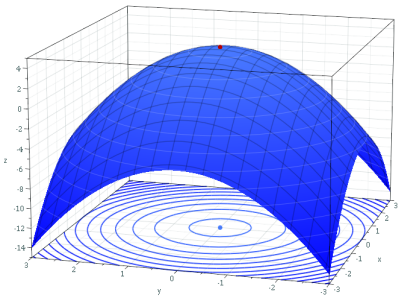

Figure 3.3: Optimizing a function (Source: http://goo.gl/I9Vba)

- Computing the gradient of f

$$\bigtriangledown f(x) = (\frac{\partial}{\partial x_k} f(x))_n \tag{3.10}$$

For the particular case of (3.9), which is simplified to a form that makes the primitive computing easier, we have:

$$\sum_{i=1}^{4} f(p) = (||p_i - p||^2 - d_i^2)^2 \tag{3.11}$$

$$\sum_{i=1}^{4} f(p) = [(p - p_i)^T (p - p_i) - d_i^2]^2 \tag{3.12}$$

$$\sum_{i=1}^{4} f(p) = [(p_x - p_{ix})^2 + (p_y - p_{iy})^2 - d_i^2] \tag{3.13}$$

Thus, the gradient $\nabla f = [h_x, h_y]$ is given by

$$\sum_{i=1}^{4} h_{x_i} = 4(p_x - p_{ix})(-(d_i.^2) + (p_{ix} - p_x)^2 + (p_{iy} - p_y)^2) \tag{3.14}$$

$$\sum_{i=1}^{4} h_{y_i} = 4(p_y - p_{iy})(-(d_i.^2) + (p_{ix} - p_x)^2 + (p_{iy} - p_y)^2) \tag{3.15}$$

- A vector d is a descent for f at x if the angle between the f gradient and d is larger than π, which means that :

$$\bigtriangledown f(x)^T d < 0 \tag{3.16}$$

- Direction of steepest descent:

$$d = -\bigtriangledown f(x) \tag{3.17}$$

Gradient Descent algorithm used (3.4):

(1)	choose start vector $x^1 \in M,\ k = 1,$ tolerance ε
While	$\left\|\nabla f\left(x^k\right)\right\| > \varepsilon$
$(k1)$	choose descent direction d^k
$(k2)$	choose step length $\alpha_k > 0$ such that $f\left(x^k\right) > f\left(x^k + \alpha_k d^k\right)$
$(k3)$	$x^{k+1} = x^k + \alpha_k d^k$
	$k = k + 1$
end	
	x^k is approximate solution of (UP)

Figure 3.4: Gradient Descent algorithm

One thing that is of most importance is the picking of the initial condition for the OM. When implementing this, it became obvious that if a wrong or inappropriate initial condition was set the program would never converge to its optimal solutions rendering the entire algorithm useless. For instance, a local minimum can appear and the solution converges to the wrong result.

3.3.2.2 Newton's Method

The NM has the advantage (when it converges) to have a quadratic convergence. The main idea is to select the descend direction according to which,

The d_k becomes:

$$d_k = -[\bigtriangledown^2 f(x_k)]^{-1} \bigtriangledown f(x_k) \tag{3.18}$$

where $\nabla^2 f$ denotes the Hessian.

This algorithm has a better stop criterion than GDM, which just considered the number of iterations. To solve this issue, another criterion is added. Now, if the error is of a set value the program stops, meaning we have found a solution that is close enough to the correct value.

$$\lambda(x) = \sqrt{\bigtriangledown f(x)^T [\bigtriangledown^2 f(x)]^{-1} \bigtriangledown f(x)} \tag{3.19}$$

$$\lambda^2(x) < \varepsilon \tag{3.20}$$

Where λ is the stop criterion.

The algorithm followed in this method is the one that follows 3.5. The only change made, however, were the formulas mentioned above.

1.	initialization	▶	choose $x_0 \in \text{dom } f$ and tolerance $\epsilon > 0$
2.		▶	choose $0 < \alpha < 0.5$ and $0 < \beta < 1$ (Armijo)
3.		▶	set $k = 0$
4.	loop	▶	compute $g_k = \nabla f(x_k)$
5.		▶	compute $H_k = \nabla^2 f(x_k)$
6.		▶	solve the linear system $H_k d_k = -g_k$
7.		▶	if $g_k^\top H_k^{-1} g_k = -g_k^\top d_k < \epsilon$ stop
8.		▶	$t = 1$
9.		▶	while $f(x_k + td_k) > f(x_k) + t\alpha g_k^\top d_k$
			$t \leftarrow \beta t$
10.		▶	$x_{k+1} = x_k + td_k$
11.		▶	$k \leftarrow k+1$ and return to loop

32

Figure 3.5: Newton's Method algorithm

After some research, the best method to use in this case is neither one of those. To obtain the best results we must use a mixed version of the two. As the NM is the fastest to converge and works with less iterations, it is considered a better method. However, if the matrix of the second primitive is singular, or close to it, it becomes impossible to use this method as we need to invert this matrix in order to implement the method. So, basically, what we do in this optimization is begin by checking if we can use the NM, and use it if possible (matrix singularity), and if not use the GDM; which works in both cases as it doesn't use matrix inversions.

In the end, the OM in this case is not that important as the results converge fast enough with each method. What becomes the main factor in locating the AUV is the trajectories involved, as will be explained further on.

3.4 Kalman Filter

3.4.1 About the filter

The Kalman Filter is an algorithm that operates recursively, using a series of measurements that contain noise, in order to produce a close-to-optimal estimation.

It uses dynamic models (for example motion equations).

The KF has several advantages when compared to other types of filters,this advantages were crucial to the picking of this type of filter:

- It works recursively (historic is not needed);
- It compensates systematic errors, adapting to alterations in the model (this is particularly important because this system has a lot of possible noise inputs, and also errors associated with measures, discretizations and such) Strength against disturbances in the sensors' outputs;
- It enables the estimation of the complete state of one system, even if not all the variables have been measured.

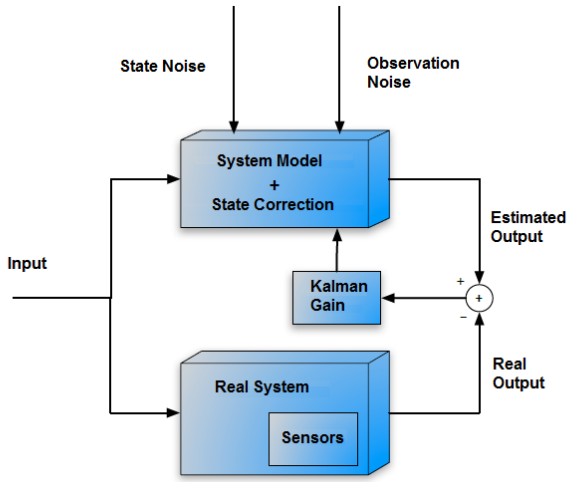

Figure 3.6: Kalman filter block diagram

In Figure 3.6 is represented the functioning of the Kalman Filter, which acts as a parallel system to the one we are using, receiving data with noise and sometimes without complete information. It then proceeds to estimate the values pretended in the respective system.

The process is represented in the generic space state model as:

$$x_{k+1} = Ax_k + Bu_k + Gw_k \tag{3.21}$$

With the measurements and observations, y:

$$y_k = Cx_k + v_k \tag{3.22}$$

Where:

- x_k - represents the state of the system, which we want to estimate;
- u_k - represents the input of the optional control;

- y_k - represents the measures obtained;
- A - is the dynamic matrix that relates the next state of the system $k+1$ with k.
- B - is the input matrix that relates the input of the optional control with the state x;
- C - is the observation matrix and relates the state with the measures obtained y_k;
- G - is the matrix that models the way the noise affects the system;
- w_k - is a random variable that represents the noise in the process;

Kalman Gain

$$K_k = P_k^- C^T (C P_k^- C^T + R)^{-1} = \frac{P_k^- C^T}{C P_k^- C^T + R} \tag{3.23}$$

- K - Kalman gain;
- P - Covariance posteriori;
- R - Covariance noise.

3.4.2 Kalman filter implementation in this project

After completing the first part of the algorithm we could observe that the trilateration worked, but only without the current's velocity. Due to the nature of this project, a navigation system that works under those circumstances is almost useless as AUVs always face some kind of current, which affects their motion.

To solve this problem we are going to use a KF, which will estimate the ocean current and also filter the AUVs position. There are many ways of designing KF, ones more suitable than others as some research was needed to understand more about the implementation of such filters [21].

To do this we will consider the ocean current constant and consider the following equations to model the system:

$$\dot{p} = v_r + v_c + \xi \tag{3.24}$$

$$\dot{v_c} = 0 + \eta \tag{3.25}$$

After having the model system defined in the previous expressions it was needed to implement it to a form more suitable of applying the KF. For this, the Euler approximation was applied resulting on the expressions that follow which are ready to be implemented in Matlab and be part of the filter's algorithm.

The formulas that represent the system are presented below:

$$\begin{bmatrix} x \\ y \\ v_{cx} \\ v_{cy} \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & \Delta t \end{bmatrix} \begin{bmatrix} x \\ y \\ v_{cx} \\ v_{cy} \end{bmatrix}_k + \begin{bmatrix} \Delta t & 0 \\ 0 & \Delta t \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_{rx} \\ v_{ry} \end{bmatrix} + \xi_k$$

$$\begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}_k \begin{bmatrix} x \\ y \\ v_{cx} \\ v_{cy} \end{bmatrix}_k + \eta_k$$

- (x, y) - vehicle position;
- (v_{cx}, v_{cy}) - ocean current velocity;
- (v_{rx}, v_{ry}) - water relative vehicle's velocity;
- $\bar{p}(\bar{x}, \bar{y})$ - position measurement;
- $\hat{v}_c$ - estimative of v_c provided by KF;
- ΔT - time step;
- ξ_k and η_k are assumed to be discrete stationary, Gaussian, zero mean white noise processes and mutually independent.

Figure 3.7 represents, in a block perspective, the different subsystems involved in this SBN system. Subsystem 1 represents the trilateration process, which already "works" in itself but is not at all reliable, needing to be filtered and completed by Subsystem 2, in which the current's speed enters and the approximation of the position already computed in the previous block. The disturbances represent noise and other errors associated with the process. The output of Subsystem 2 will provide feedback to the system.

The first (Trilateration) and second (Kalman) block (subsystems) are intimately related as they share the use of some values. For example, q, computed on the first block depends on the output of the second one (this is due to the need of the ocean's current for the position estimation computing) and this block, the Kalman one, uses the results of the trilateration, taking in consideration the values of the current and filtering the noise in order to make possible the conversion of the algorithm.

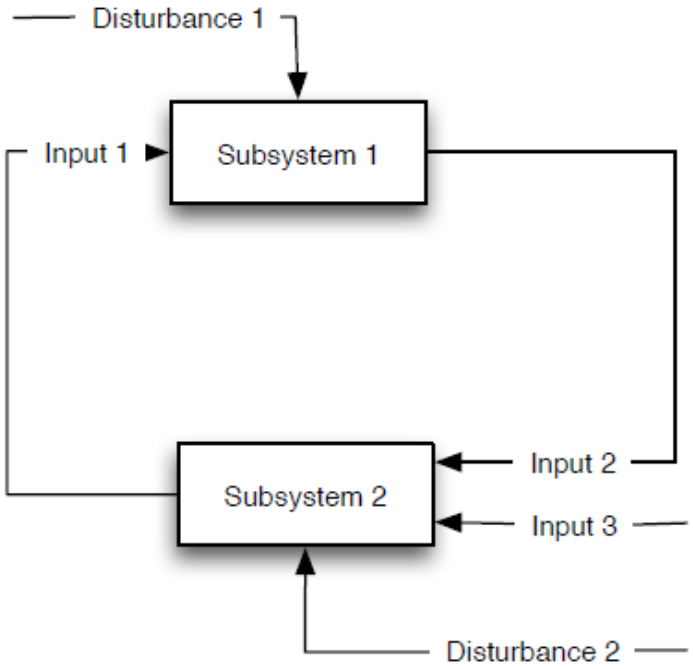

Figure 3.7: Algorithm block diagram (Source: [2])

Chapter 4

Cooperative Navigation

This chapter addresses the cooperative navigation for multiple marine robotic vehicles. The main idea is to make use of the fact that each individual member of the group could benefit from navigation information obtained from other members. For underwater vehicles cooperative navigation is considerably more challenging but very attractive. Only few vehicles are needed to maintain an accurate estimate of their positions through sophisticated (and expensive) navigation sensors. The other ones can have less sophisticated navigation. [22].

In this chapter, by extending the concept of single beacon navigation for one AUV, we focus in two types of CN:

- The one between an ASC and one or more AUVs;

- The one between various AUVs.

The first one has obvious advantages when it comes to navigation. As there is a ASC located in the sea level, at the surface, it enables the use of GPS to locate the position of the surface, making easier to locate the area where the related AUVs are exploring. These vehicles use, then, another navigation algorithm to compute their location underwater using ASs, in a trilateration system that uses a net of beacons and obtains the location of the vehicle in relation to the ASC. See Figure 4.1.

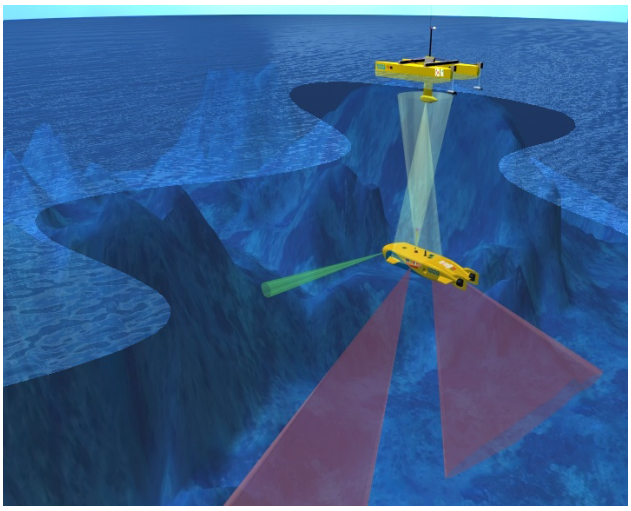

Figure 4.1: Cooperative navigation between ASC and AUV (Source: [1])

The second one is used in order to have a permanent communication between AUVs which enables the possibility to explore in formation to cover more area and achieve better goals. Another possibility of this CN technique is to make only one of the vehicles to transmit and receive signals to the beacon system and use this navigation information to control the navigation of the rest of the fleet. See Figure 4.2.

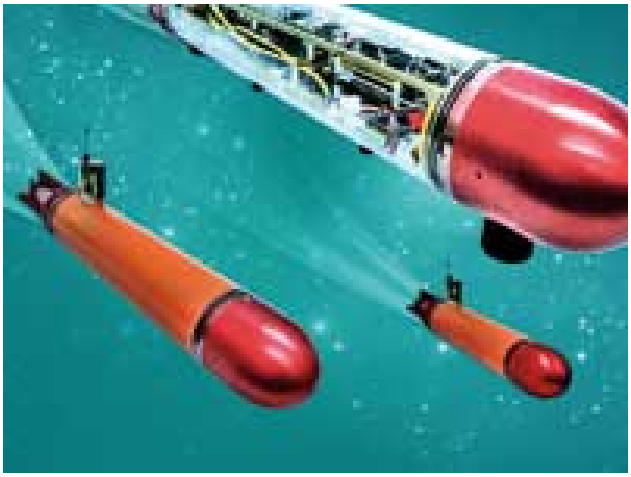

Figure 4.2: Underwater fleet navigation (Source: http://goo.gl/D9CJ4)

4.1 CN between AUV and ASC

This type of CN is very attractive because the idea is to use an ASC, which is floating on the water, so above sea level, and for this reason it becomes possible to know its coordinates through the GPS system. The setup proposed is to make the ASC to play the role of the beacon as in the case of Chapter 3. However, in this setup, we will have a moving beacon, as the ASC will be able

to move and stay still. Moreover, the ASC can in principle make desirable selected movements to allow a better performance on the computation of the position of the AUV, which is working in exploration, rescue, reconnaissance or such missions. By this we get the position of the AUV related to the ASC, which we have previously located through GPS.

To extend the SBN to the cooperative setup, some modifications in the algorithm are needed. In particular, the computation of the location of the VBN has to be modified by taking into account the displacement of the beacon. More precisely, in our setup with four (virtual) beacons (N=4) we have

$$p_4 = p_{beacon}(12) \tag{4.1}$$

$$p_3 = p_4 + int_{v34} - [p_{beacon}(12) - p_{beacon}(8)] \tag{4.2}$$

$$p_2 = p_3 + int_{v23} - [p_{beacon}(8) - p_{beacon}(4)] \tag{4.3}$$

$$p_1 = p_2 + int_{v12} - [p_{beacon}(4) - p_{beacon}(0)] \tag{4.4}$$

- p(0) - position of the moving beacon at the instant $t = 0s$;
- p(4) - position of the moving beacon at the instant $t = 4s$;
- p(8) - position of the moving beacon at the instant $t = 8s$;
- p(12) - position of the moving beacon at the instant $t = 12s$;

4.2 CN between AUV and AUV

For different purposes it becomes an advantage to use an AUV as a beacon reference for other AUVs. This can be used, for instance, to guide multiple vehicles in fleet formation for reconnaissance or rescue purposes. The other vehicles use one AUV as their beacon and can use it as a reference to take their right positions. In environments where few beacons are placed this could have an obvious role and it can also be used mixed with the cooperation with ASCs depending on the purpose we are aiming to.

As this type of navigation involves two or more AUVs, we cannot use the same algorithm as with the previous form of cooperation between systems. This is due to the fact that we cannot rely on the GPS, positioning system as both AUVs will be underwater.

As the position of both vehicles is unknown, and one of them will be the beacon for the other one, we will have to compute the displacement of the vehicle that will serve as the beacon in a similar process as we did before when we implemented the algorithm for one AUV.

Once again, what needs to be changed is how the virtual net beacon is generated and therefore, the calculations of their positions.

The integral of the speed of the beacon-AUV will be computed in order to get its respective displacement and later add it to each according P.

$$p_4 = p_{beacon}(12) \tag{4.5}$$

$$p_3 = p_4 + int_{v34} - [p_{beacon}(12) - p_{beacon}(8)] \tag{4.6}$$

$$p_2 = p_3 + int_{v23} - [p_{beacon}(8) - p_{beacon}(4)] \tag{4.7}$$

$$p_1 = p_2 + int_{v12} - [p_{beacon}(4) - p_{beacon}(0)] \tag{4.8}$$

In this case, the AUV with the beacon needs to have more sophisticated inertial sensors in order to integrate its velocities without accumulating much error.

Chapter 5

Results and discussion

5.1 Single Navigation

In this part, the results obtained throughout the SBN algorithm implementation will be discussed. Some details that should be mentioned about the simulation process are:

- Every 12 seconds, we update the Kalman filter with a new location.
- The sampling time of the overall algorithm is 0.1 s. This time is defined in the configuration parameters.
- It is implicitly assumed that the velocity measurements are obtained at each sampling time. The range measurements are obtained every 4 s.
- For this simulations, we run the program 800s, which was a time we considered appropriate for the AUV to travel enough as to make it easier to be tracked.

For simulation purposes, the path picked for the AUVs movement is a relatively common one. It contains many of the standard types of movements the AUV can have, turning left and right and heading forward and is also realistic as it is the type of paths these vehicles follow when exploring and other similar types of missions.

It is important to stress that several paths were tested during and after implementing the program because the ability to track the right position and for the algorithm to converge it could be concluded that there are some paths that are better than others. This will be shown clearly in the next figures.

For example, a path that may have some problems is when when the vehicle moves in a straight line away from the beacon and turns out that in this case the location of the positions of the (virtual) beacons are aligned in that a line parallel to the trajectory. In this case, the algorithm may diverge as there is no way of knowing the real distance between the beacon and the vehicle; we just know it is somewhere along that line.

For this reason, as we will see further on, when the navigation is cooperative, with the aid of a moving AUV or ASC, we get better results as we can avoid the greater flaws of this system and by moving the beacon in some "good" directions.

After realising this, some paths were set for the AUV to follow. As expected some with better results than others, but none yet having the best desired results, which are only achieved in cooperation.

5.1.1 Straight Line Movement

Obtained by simply putting the angular speed to zero and keeping the linear speed at 1 m/s.

5.1.1.1 Without noise and without current

To start the simulation of the results of the program implemented in Matlab it was wise to start with the simplest of the cases. In this simulation we can observe the movement made by the vehicle which is coloured in green and we can compare it with the estimated movement obtained by superposing the values of the estimated positions(black dots) of the vehicle using the trilateration algorithm and the Kalman response, given in red. See Figure 5.1.

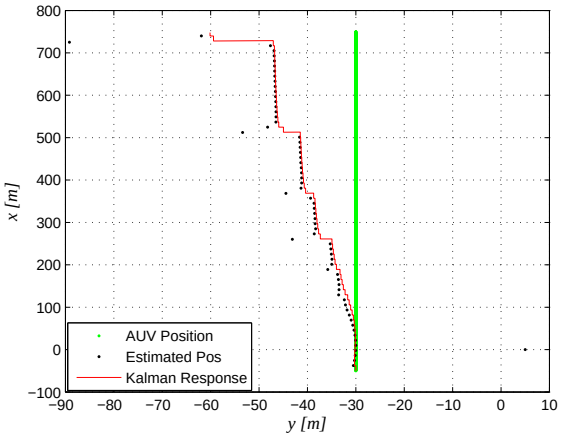

Figure 5.1: Line movement without noise or current

5.1.1.2 With noise and without current

After the previous simulation we add some noise to make this simulation more realistic since, when tested in the real world these kind of systems always involve noise. This happens in the sensors , under influence of the environment (storms or such), errors in calculations, so adding it up helps considering these factors. See Figure 5.2.

This problem is easily solved by the filter and does not introduce too much harm to the functioning of the trilateration.

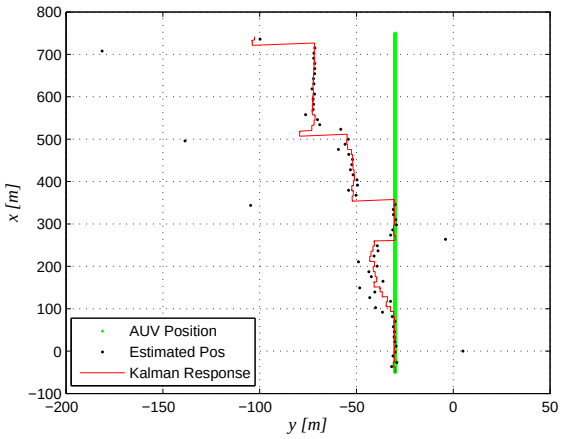

Figure 5.2: Line movement with noise and without current

5.1.1.3 With current and without noise

Firstly the testing was done without considering the current speed. Later it had to be implemented as it is a crucial factor in this field.

The currents affect greatly the vehicle trajectory and if we don't consider them in the algorithm, it will not work as it diverges and does not bring any good results.

To be able to compute the position of the vehicle with current speed it is imperative that we use the KF. Due to its capability to act recursively, the KF is used to estimate the current disturbance and use this estimate in order to find the correct location of the vehicle. See Figure 5.3.

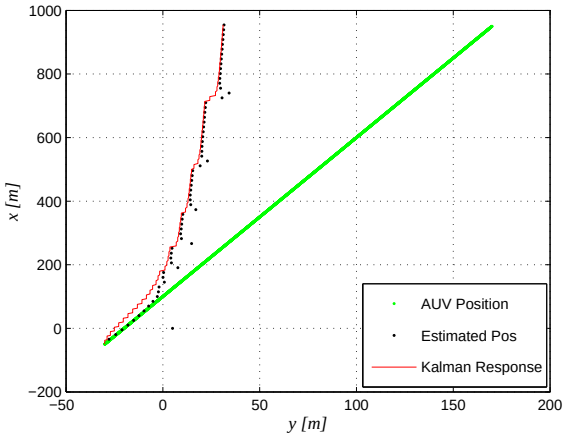

Figure 5.3: Line movement with current and without noise

5.1.1.4 With current and noise

Once again, after the program is tested with current speed we must include noise for realism purposes. See Figure 5.4.

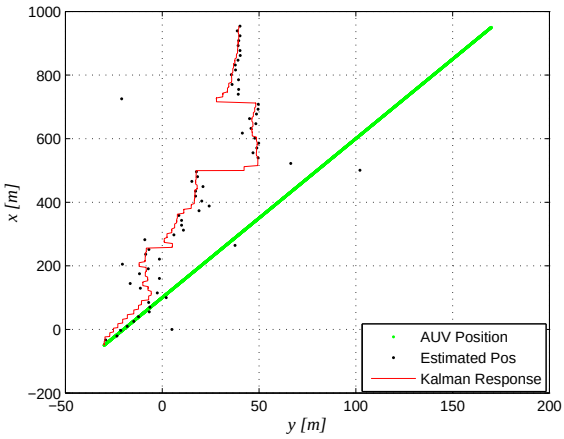

Figure 5.4: Line movement with current and noise

5.1.2 Circular Movement

For this movement we keep the same linear speed and give the angular speed a value, in this case 0.1 rad/s.

5.1.2.1 Without noise and without current

The circular motion is better than the linear one because it moves in more directions and is less likely to stay in a "blind angle zone". See Figure 5.5.

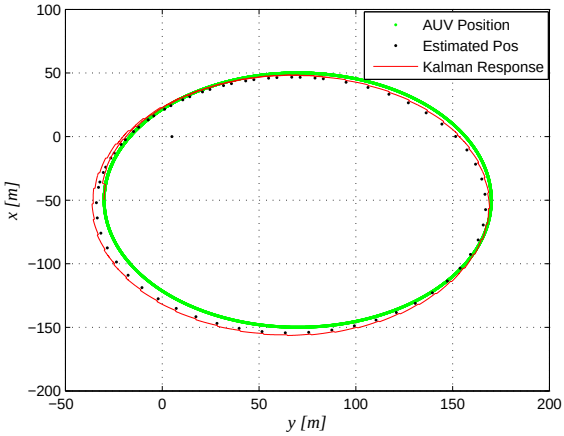

Figure 5.5: Circular movement without current or noise

It presents a good behaviour when it's not in the presence of current or noise.

5.1.2.2 With noise and without current

With noise, it still has the appropriate response, converging well, just with some minor deviations as it is supposed to have. See Figure 5.6.

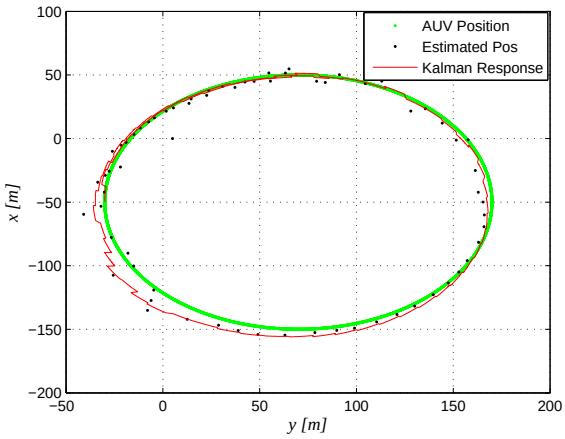

Figure 5.6: Circular movement with noise and without current

5.1.2.3 With current and without noise

When in the presence of current, the movement of the AUV changes, and is no more a perfect circle motion. It still manages to track the position, with a little deviation in the curve which is tight for the few dots we plot. See Figure 5.7.

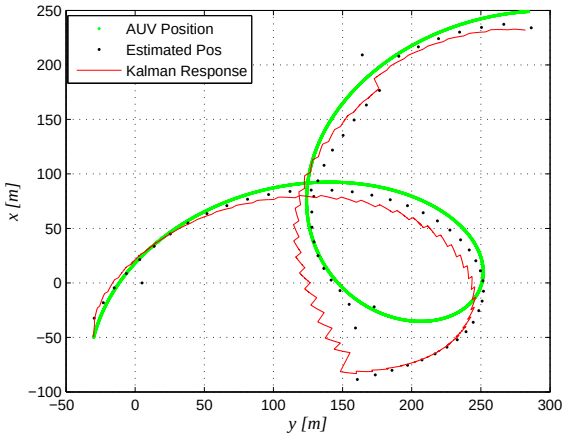

Figure 5.7: Circular movement with current and without noise

5.1.2.4 With current and noise

With noise, we obtain similar results. See Figure 5.8.

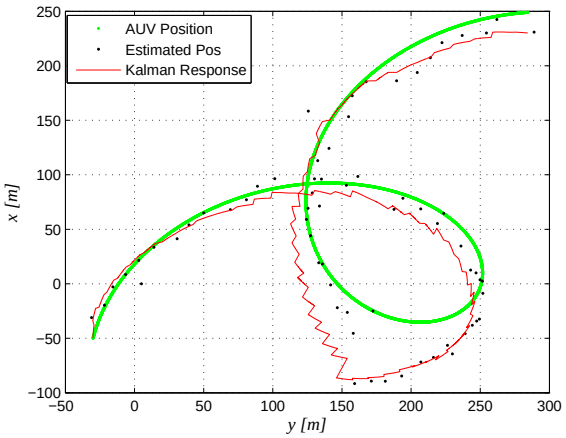

Figure 5.8: Circular movement with current and noise

5.1.3 S Movement

This movement is more involved. As we wanted it to go straight for a distance and then curve for a while before going straight again and curve the other way and repeat the motion; we had to implement some code that would create the desired output signal. This code can be found in the appendix.

5.1.3.1 Without noise and without current

In this case the output of the KF tries to follow the movement of the AUV but without much success. It meets paths that are on the blind angle and gets lost in the curves. Clearly calls for a better way to track it, this is solved with a cooperating vehicle, a beacon that has movement. See Figure 5.9.

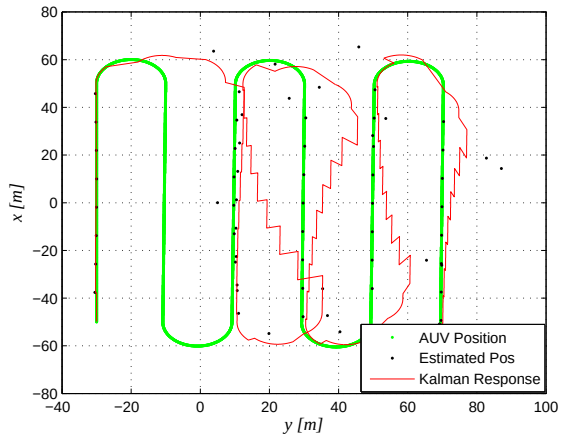

Figure 5.9: S movement without current or noise

5.1.3.2 With noise and without current

Even worse results, as expected. See Figure 5.10.

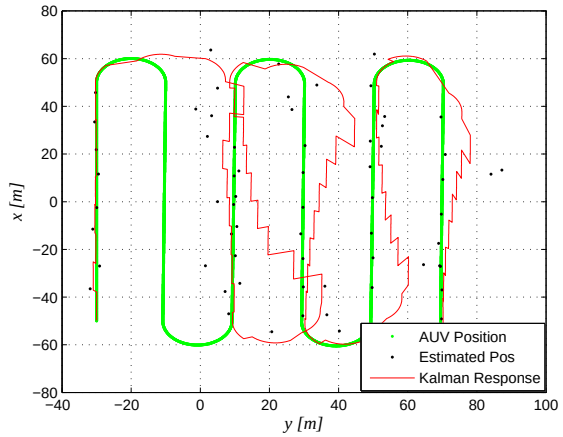

Figure 5.10: S movement with noise and without current

5.1.3.3 With current and without noise

It is peculiar how the current speed helps in the convergence of the algorithm. In fact, by giving it another motion, it drives it away from the movements that are harder to track. See Figure 5.11.

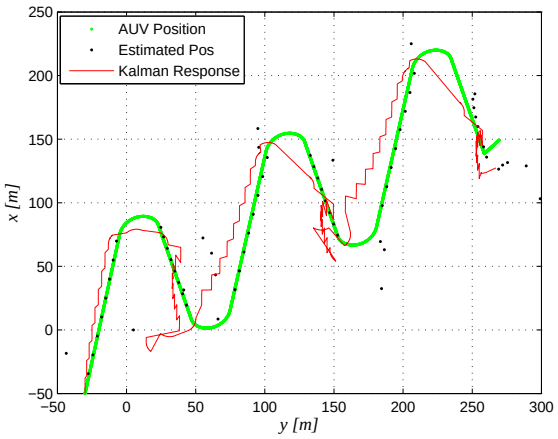

Figure 5.11: S movement with current and without noise

5.1.3.4 With current and noise

Similar results, but in this simulation the noise helped in some parts of the movement and worsened the others. See Figure 5.12.

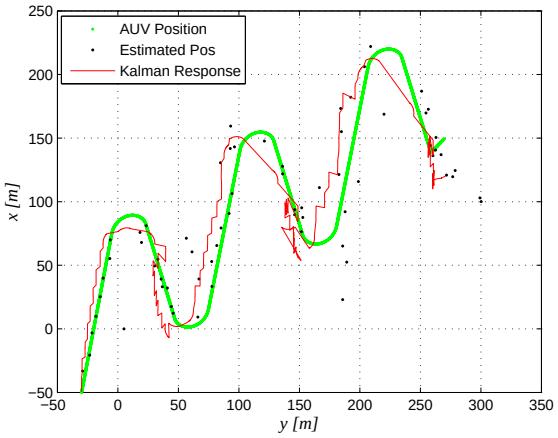

Figure 5.12: S movement with current and noise

5.2 Cooperative Navigation - Between ASC and AUV

For the reasons stated before, the functioning of this algorithm and its implementation in reality is vastly improved with the aid of a moving device, that avoids its weak points. This device in this case will be an ASC, moving in the surface of the ocean, in a movement that is tested by the following simulations as to see what would be a good choice for a ever converging position estimations of the AUV. The movement of the AUV stays the same in every simulation because the S movement is the one used in most operations(reconnaissance, exploration, etc). The reason

we tested so many movements in the SN was to better understand what the results would be and in which cases it would have better and worse results.

5.2.1 Straight Line Movement of the ASC

In a research process we tried different movements for the ASC, to find out what would be the best one to use when implementing the program in real vehicles.

5.2.1.1 Without noise and without current

As expected, in a straight line there are many spots that are hard to track, so it gives a poor result. See Figure 5.13.

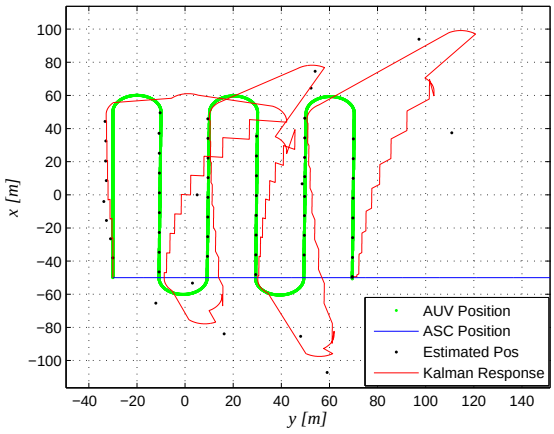

Figure 5.13: Line movement without current or noise

5.2.1.2 With noise and without current

With noise, we get a similar result.See Figure 5.14.

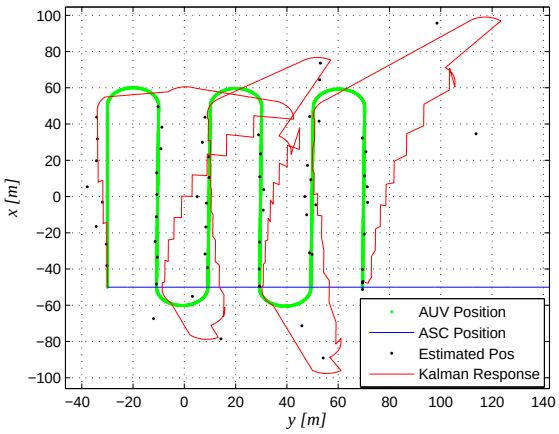

Figure 5.14: Line movement with noise and without current

5.2.1.3 With current and without noise

Once again, the current actually improves the result. See Figure 5.15.

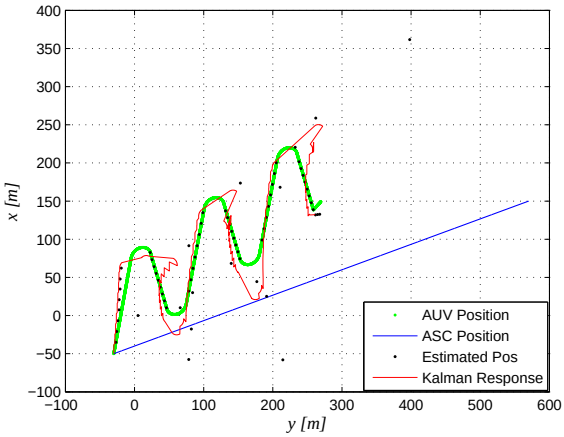

Figure 5.15: Line movement with current and without noise

5.2.1.4 With current and noise

And the noise causes problems in the curves. See Figure 5.16.

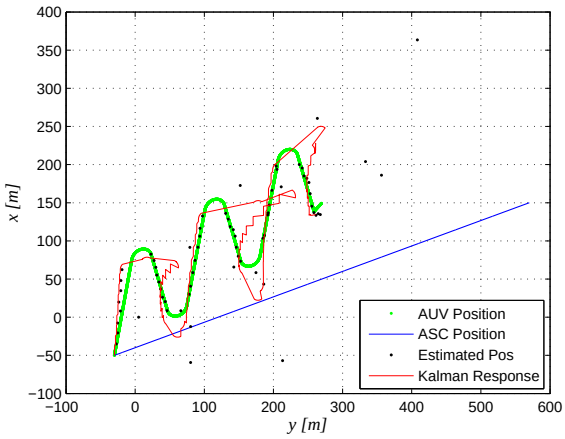

Figure 5.16: Line movement with current and noise

5.2.2 Circular Movement of the ASC

With a circular movement we get better results than with the straight line, but these are still not the results we are after.

5.2.2.1 Without noise and without current

Still with large deviations, hard to converge. See Figure 5.17.

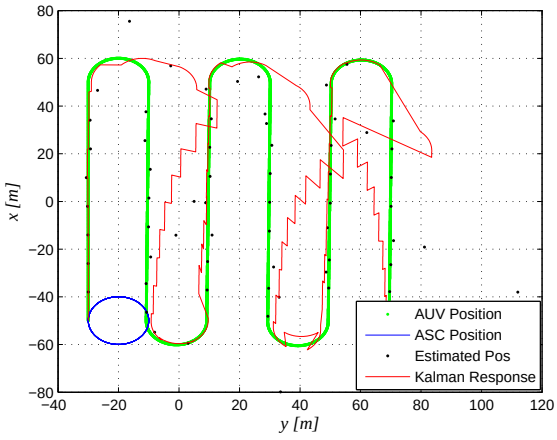

Figure 5.17: Circular movement without current or noise

5.2.2.2 With noise and without current

Similar to the previous simulation. See Figure 5.18.

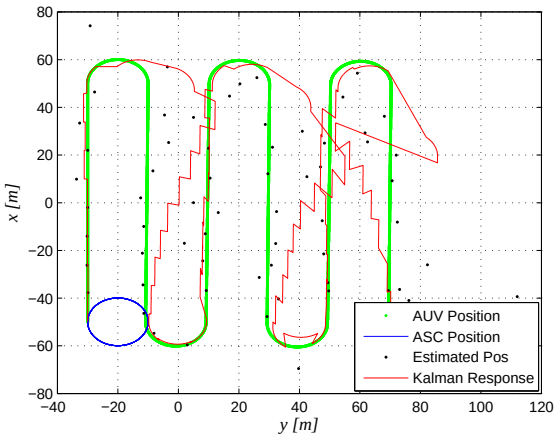

Figure 5.18: Circular movement with noise and without current

5.2.2.3 With current and without noise

Current helps the algorithm convergence, giving better results. See Figure 5.19.

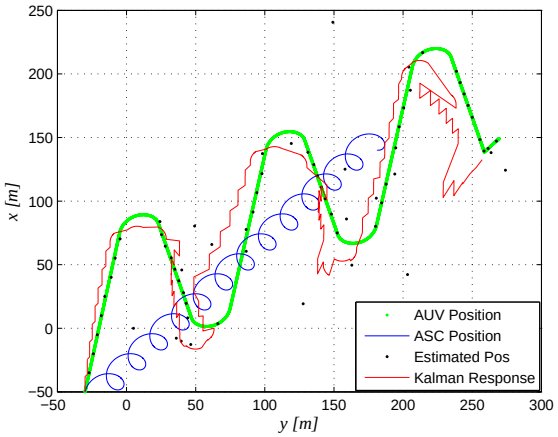

Figure 5.19: Circular movement with current and without noise

5.2.2.4 With current and noise

Noise does not affect much or, if anything, even makes it better. See Figure 5.20.

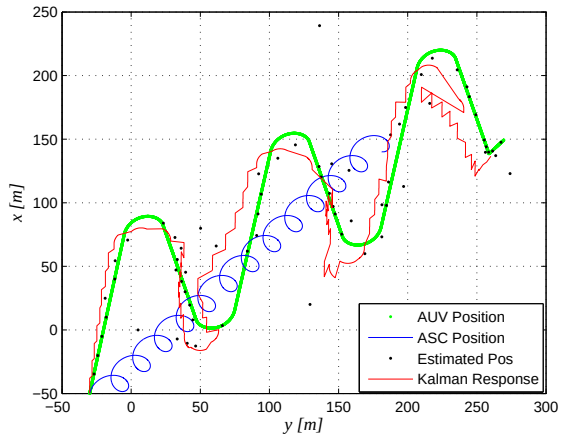

Figure 5.20: Circular movement with current and noise

Just to show the results with a bigger circular movement. See Figure 5.21.

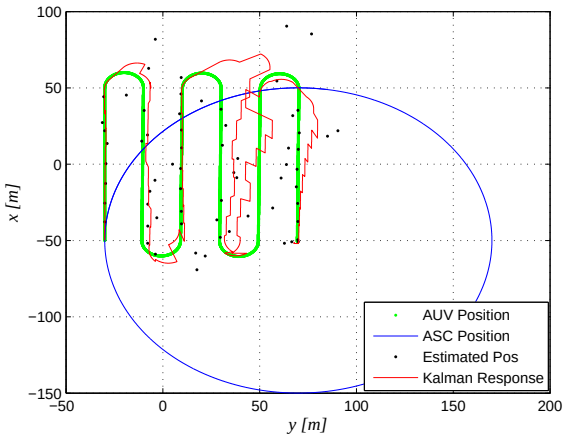

Figure 5.21: Circular movement with current and noise

5.2.3 S Movement of the ASC

In this case, the ASC follows the AUV right above it, on the surface. This is the movement we were looking for.

5.2.3.1 Without noise and without current

Perfect results. See Figure 5.22.

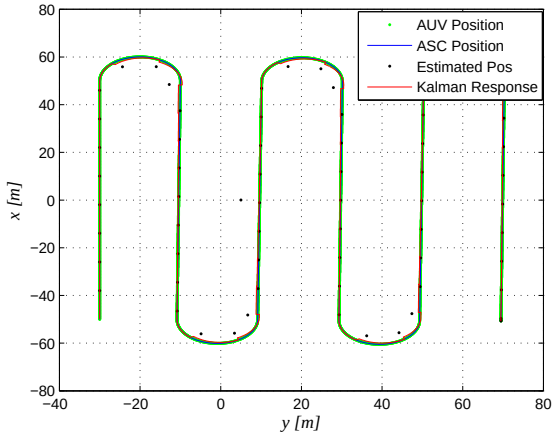

Figure 5.22: S movement without current or noise

5.2.3.2 With noise and without current

Noise is hardly noticeable. See Figure 5.23.

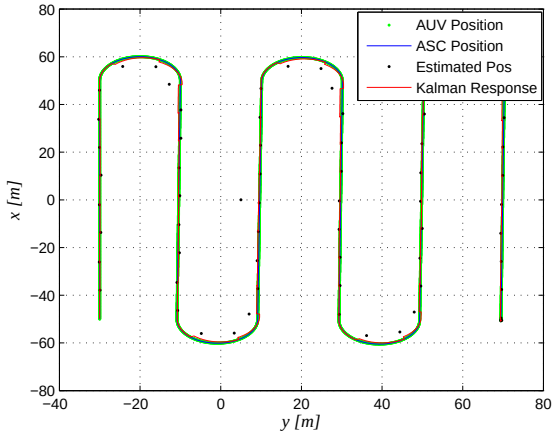

Figure 5.23: S movement with noise and without current

5.2.3.3 With current and without noise

Tracks well the AUV under the influence of the current. See Figure 5.24.

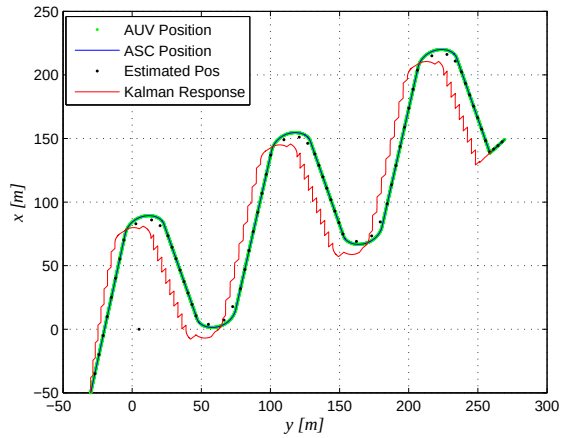

Figure 5.24: S movement with current and without noise

The next extra figure shows what happens when the vehicle wants to turn back against the current, and has the same speed. As expected the vehicle can't move against it. See Figure 5.25.

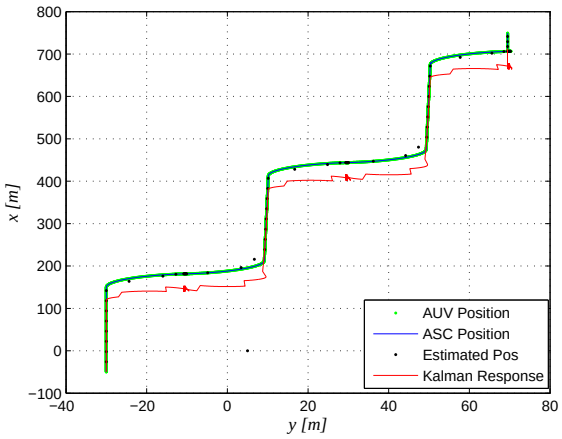

Figure 5.25: S movement with current and without noise

5.2.3.4 With current and noise

Noise does not affect this performance at all.See Figure 5.26.

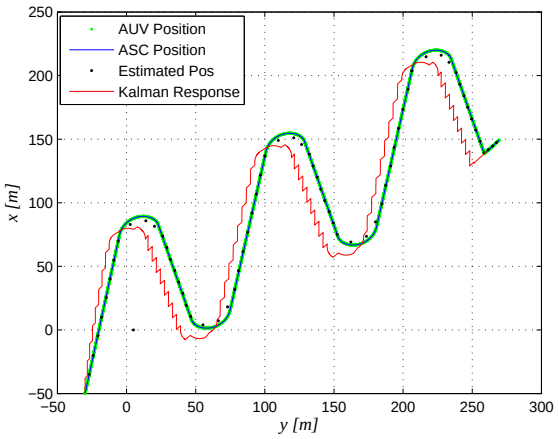

Figure 5.26: S movement with current and noise

5.2.4 N Movement of the ASC

This motion was just tested for curiosity, by far not appropriate. Too many diverging points.

5.2.4.1 Without noise and without current

Poor performance. See Figure 5.27.

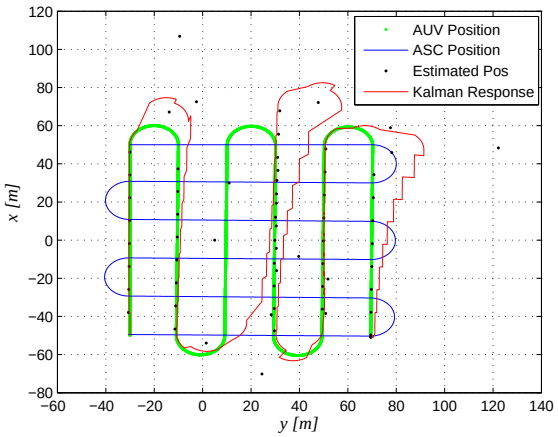

Figure 5.27: S movement without current or noise

5.2.4.2 With noise and without current

See Figure 5.28.

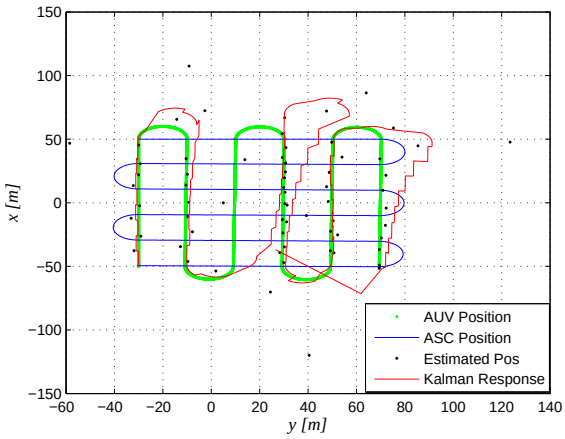

Figure 5.28: S movement with noise and without current

5.2.4.3 With current and without noise

Improves significantly with the current. See Figure 5.29.

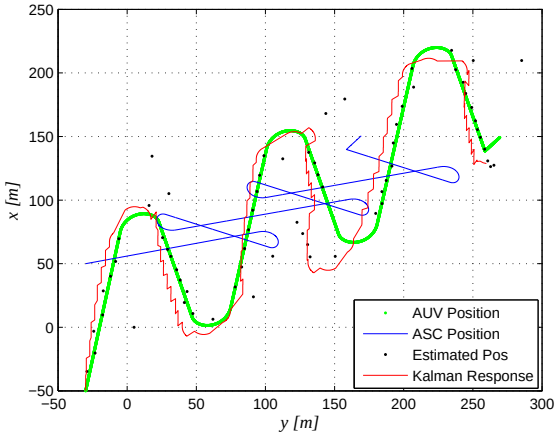

Figure 5.29: S movement current and without noise

5.2.4.4 With current and noise

Highly affected by noise. See Figure 5.30.

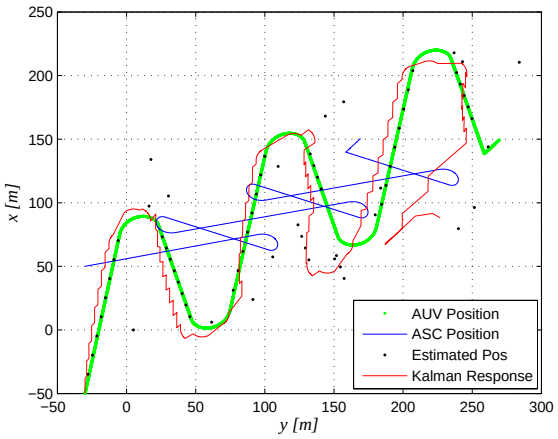

Figure 5.30: S movement with current and noise

We conclude that, as we had already foreseen in calculations and reasoning that the best motion for the ASC would be above the AUV.

Some more results are added in Appendix 3, related to velocities obtained and when noise is added to them.

5.3 Cooperative Navigation - Between AUV and AUV

In the previous chapter we already explained the main difference in the implementation of this system. However, we can't fully replicate the conditions behind this functioning as the computing would be done in the beacon AUV (integration of its speed) and then sent to the AUV we want to locate. In this case, for easier implementation and just to have some idea of what would happen in the AUV to AUV case, the implementation was done in the same block (which means the same AUV) as the rest. The difference, however, would not be that big, having to consider some delay and error, in fact, but still some approximation of the reality is possible.

It does not make sense to print all simulations again as the results are close to the ones represented above.

Chapter 6

Conclusions and Further Research

6.1 Summary

To sum up, navigation algorithms together with better and new sensors are continuously evolving at an increasing rate, which is why we cannot yet foresee how it will unravel in the far future.

Nowadays and in the near future, navigation is an important role and is constantly evolving with the objective of consuming fewer resources and granting better results. In this work we addressed the way to use fewer beacons in the trilateration process used underwater to track the location of AUVs. This was possible through the use of advanced navigation algorithms. This is one particular case where better technology/knowledge leads to better and cheaper performance.

For the algorithm proposed we have realized that the use of a KF was essential, otherwise we could only compute the location of the AUV without ocean current, which would limit considerably its application.

The cooperative component is ever becoming more of a must in the evolution of navigation as it unlocks many possibilities. It also is economically viable as it is a way to spare equipment; by using a ASV/AUV as a beacon and give the whole operation more mobility and more effectiveness. Large systems composed by ASC and multiple AUVs all navigating in cooperation can fulfil many important roles.

6.2 Further Research

The next immediate step of this project would be to convert the Matlab code to C code and download it to AUVs in order to test it in a controlled environment at first (pool) and later send it on missions, always collecting the data from the missions in order to perfect the code and algorithm used.

Appendix A

Appendix 1

A.1 Schematics and schemes

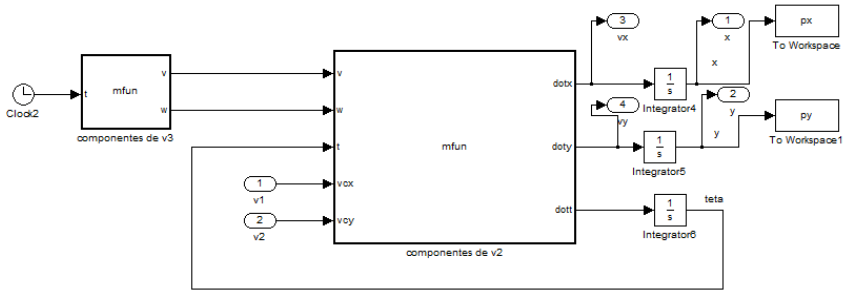

Figure A.1: AUV block

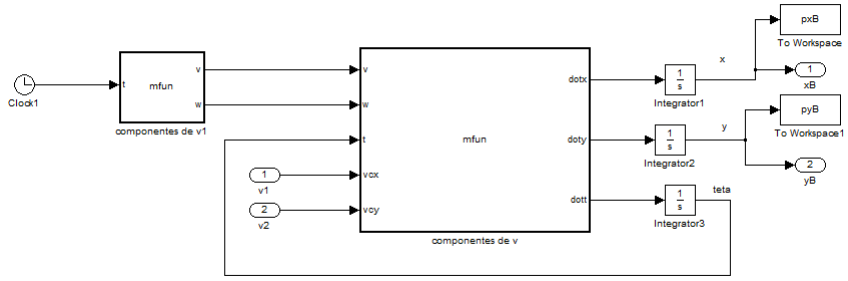

Figure A.2: ASC Beacon

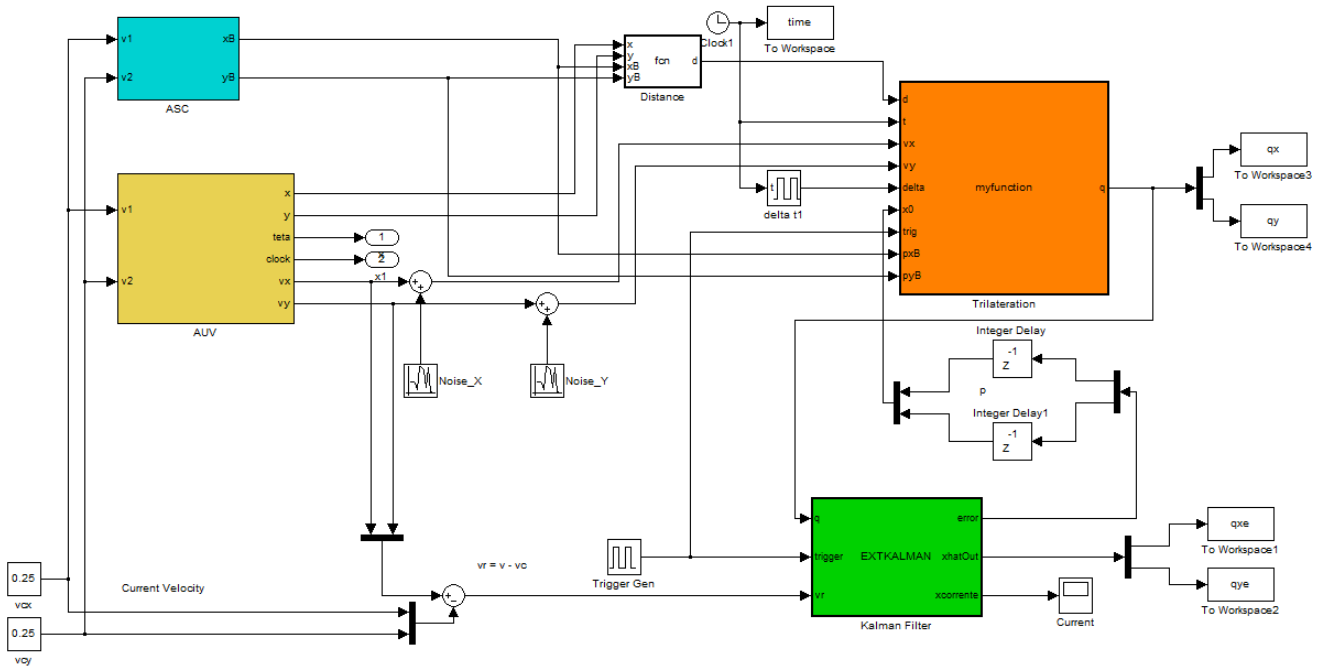

Figure A.3: Cooperative ASC and AUV

A.2 MatLab blocks and modules code

Plot

```
1
2 figure(1)
3 plot(py , px,'g.',pyB,pxB, 'b-',qy ,qx ,'k.', qye , qxe,'r');
4 xlabel('y [m]','FontName','times','Fontsize',12,'FontAngle','Italic');
5 ylabel('x [m]','FontName','times','Fontsize',12,'FontAngle','Italic');
6 grid
```

Distance Calculation

```
1
2 function d = distance(x,y,xB,yB)
3 d = sqrt ( (x-xB)^2 + (y-yB)^2);
```

Vehicle Speed in XX's and YY's

```
1
2 function [dotx,doty,dott] = Vspeed(v,w,t,vcx,vcy)
3 dott= w;
4 doty = v*sin(t) + vcy;
5 dotx = v*cos(t) + vcx;
```

Trilateration

```
1
2 %--FUNCTION THAT PASSES THE VELOCITIES--
3
4
5
6 function q   = myfunction(d, t, vx, vy ,?, x0,trig,pxB ,pyB)   %output ...
      and inputs
7
8 persistent i;                                                    ...
      %defining all variables
9 persistent w;
10 persistent myflag;
11 persistent myflag2;
12
13 persistent vx_array;
14 persistent vy_array;
15 persistent d_array;
16
```

```
17 persistent PB_0;   %represents the position of the moving beacon at the ...
       times stated
18 persistent PB_4;
19 persistent PB_8;
20 persistent PB_12;
21
22 persistent xi;
23 persistent x1;
24 persistent x2;
25 persistent x3;
26
27 persistent yi;
28 persistent y1;
29 persistent y2;
30 persistent y3;
31
32 persistent p;
33
34 persistent px;
35 persistent py;
36
37 persistent hx;
38 persistent hy;
39
40 persistent f;
41
42 persistent  p1x;
43 persistent  p2x;
44 persistent  p3x;
45 persistent  p4x;
46 persistent  p1y;
47 persistent  p2y;
48 persistent  p3y;
49 persistent  p4y;
50
51 persistent p1;
52 persistent p2;
53 persistent p3;
54 persistent p4;
55
56 %variables that define the displacement of the vehicle in Y from the ...
       position y1 to y2; the others follow the same logic
57
58 persistent int_vy_12;
59 persistent int_vy_23;
60 persistent int_vy_34;
61
62 %variables that define the displacement of the vehicle in Y from the ...
       position x1 to x2; the others follow the same logic
```

```
63
64 persistent int_vx_12;
65 persistent int_vx_23;
66 persistent int_vx_34;
67
68 persistent int_v_12;            %concatenates the x and y components
69 persistent int_v_23;
70 persistent int_v_34;
71
72 persistent f2;
73 persistent tk;
74
75  %fills in the variables in the beginning of the program, what is empty ...
       becomes 0
76
77 if isempty(w)
78
79     w = 0;
80     i = 0;
81     myflag = 0;
82
83     myflag2 = 0;
84
85     vx_array = zeros(120,1);
86     vy_array = zeros(120,1);
87     d_array = zeros(4,1);
88
89     PB_0 = zeros(2,1);
90     PB_4 = zeros(2,1);
91     PB_8 = zeros(2,1);
92     PB_12 = zeros(2,1);
93
94     tk=0;
95
96     xi = 0;
97     x1 = 0;
98     x2 = 0;
99     x3 = 0;
100
101    yi = 0;
102    y1 = 0;
103    y2 = 0;
104    y3 = 0;
105
106    px=0;
107    py=5;
108
109    p1x=0;
110    p2x=0;
```

```
111      p3x=0;
112      p4x=0;
113      p1y=0;
114      p2y=0;
115      p3y=0;
116      p4y=0;
117
118      f = zeros(2,1);
119
120      p  = zeros(2,1);
121      p1 = zeros(2,1);
122      p2 = zeros(2,1);
123      p3 = zeros(2,1);
124      p4 = zeros(2,1);
125
126      int_vy_12 = 0;
127      int_vy_23 = 0;
128      int_vy_34 = 0;
129
130      int_vx_12 = 0;
131      int_vx_23 = 0;
132      int_vx_34 = 0;
133
134      int_v_12 = zeros(2,1);
135      int_v_23 = zeros(2,1);
136      int_v_34 = zeros(2,1);
137      trigger = 0;
138  end
139   trigger = 0;
140
141  % in the beginning of the program, fills in the value of 120 ...
         velocities, which have entered in the block
142   % this velocities will be used to compute the positions; this happens ...
          with the integration of the vehicle's displacement
143
144  if (t < 0.1)
145
146      w=1;
147      myflag = 1;
148
149  % myflag is used to make sure the program only uses one velocity every ...
         0.1 seconds
150  end
151
152
153
154  if(t≥0.1 && w<121)
155      if (Δ == 1 && myflag == 1)
156      vx_array(w) = vx;
```

```
157     vy_array(w) = vy;
158 
159         if (w==1)
160         d_array(1)=d;
161         PB_0 =[pxB;pyB];
162         end
163 
164     w=w+1;
165     myflag = 0;
166 
167 %is put to 0 after writing the value of the velocities to create the ...
        Rising Edge effect
168 
169     end
170 
171 myflag=1;
172 end
173 
174 
175 
176 
177 %--WRITTING OF THE D'S--
178 
179 if(w==41)                                %every 4 seconds, saves the ...
        value of d and PB
180     t;
181     d_array(2)=d;
182      PB_4 =[pxB;pyB];
183 
184 end
185 if(w==81)
186     t;
187     d_array(3)=d;
188      PB_8 =[pxB;pyB];
189 end
190 if(w==121)
191     t;
192     d_array(4)=d;
193     PB_12 =[pxB;pyB];
194 end
195 
196 
197 d1 = d_array(1) ;
198 d2 = d_array(2) ;
199 d3 = d_array(3) ;
200 d4 = d_array(4) ;
201 
202 %--CODE THAT FOR EVERY PERIOD OF T COMPUTES THE AREA OF THE RESPECTIVE ...
        INTEGRAL--
```

```
%%{



if(w == 121)                          % after writing all velocities

    % instant from 0 to 4 seconds, calculation of the displacement and ...
        acquisition of d1
    for i=1:40


        xi = 0.1 * vx + xi;
        yi = 0.1 * vy + yi;


    end


    x1 = xi;
    int_vx_34 = x1;

    y1 = yi;
    int_vy_34 = y1;

    int_v_34 = [int_vx_34; int_vy_34];

    xi=0;
    yi=0;

   % instant from 4 to 8 seconds, calculation of the displacement and ...
       aquisition of d2

    for i=41:80


        xi = 0.1 * vx + xi;
        yi = 0.1 * vy + yi;


    end


    x2=xi;
    int_vx_23= x2;

    y2=yi;
    int_vy_23= y2;

    int_v_23 = [int_vx_23; int_vy_23];
```

```
250
251     xi=0;
252     yi=0;
253
254     % instant from 8 to 12 seconds, calculation of the displacement and ...
            aquisition of d3
255
256     for i=81:120
257
258
259         xi = 0.1 * vx + xi;
260         yi = 0.1 * vy + yi;
261
262
263     end
264
265
266     %after 12 seconds, aquisition of d4
267
268     x3=xi;
269     int_vx_12=x3;
270
271     y3=yi;
272     int_vy_12=y3;
273
274     int_v_12 = [int_vx_12; int_vy_12];
275
276     xi=0;
277     yi=0;
278
279  %--FUNCTION THAT COMPUTES THE P'S THAT WILL BE USED IN THE OPTIMIZATION--
280
281     p4 = [0;0] + PB_12;
282     p3 = p4 + int_v_34 - (PB_12 - PB_8);
283     p2 = p3 + int_v_23 -(PB_8 - PB_4);
284     p1 = p2 + int_v_12 -(PB_4 - PB_0);
285
286     p4x=p4(1);
287     p3x=p3(1);
288     p2x=p2(1);
289     p1x=p1(1);
290
291     p4y=p4(2);
292     p3y=p3(2);
293     p2y=p2(2);
294     p1y=p1(2);
295  %}
296
297  %------OPTIMIZATION FUNCTION------
```

```
298
299  w=1;
300     if (trig == 1 && myflag2 == 0)
301
302         xk=x0;
303
304
305         n=0;
306         nmax=4000;                                          %set max ...
                number of iterations
307
308         alf = 0.2;
309         bet = 0.5;
310
311         while (n<=nmax)                                    %set ...
                while-conditions
312
313
314             f =((norm (xk-p1))^2 -d1^2)^2 +((norm (xk-p2))^2 -d2^2)^2 ...
                    +((norm (xk-p3))^2 -d3^2)^2 +((norm (xk-p4))^2 -d4^2)^2;
315
316             hx=4*(px-p1x)* (-(d1.^2) + (p1x - px)^2 + (p1y - py)^2 ) + ...
                    4*(px-p2x)* (-(d2.^2) + (p2x - px)^2 + (p2y - py)^2 ) + ...
                    4*(px-p3x)* (-(d3.^2) + (p3x - px)^2 + (p3y - py)^2 ) + ...
                    4*(px-p4x)* (-(d4.^2) + (p4x - px)^2 + (p4y - py)^2 );
317
318             hy=4*(py-p1y)* (-(d1.^2) + (p1x - px)^2 + (p1y - py)^2 ) + ...
                    4*(py-p2y)* (-(d2.^2) + (p2x - px)^2 + (p2y - py)^2 ) + ...
                    4*(py-p3y)* (-(d3.^2) + (p3x - px)^2 + (p3y - py)^2 )+ ...
                    4*(py-p4y)* (-(d4.^2) + (p4x - px)^2 + (p4y - py)^2 );
319
320             gf = [ hx ; hy ] ;      % parcial primitives of g, x and y; ...
                    gradient
321             dk = -gf;
322
323             tk=1;
324
325             f2 = ((norm (xk + tk*dk-p1))^2 -d1^2)^2 +((norm (xk + ...
                    tk*dk-p2))^2 -d2^2)^2 +((norm (xk + tk*dk-p3))^2 ...
                    -d3^2)^2 +((norm (xk + tk*dk-p4))^2 -d4^2)^2   ;
326
327             aux =  f + tk * alf * (gf' * dk);
328
329
330 %computes the value of tk, which becomes smaller every iteration
331
332             while( f2 > aux(1))
333
334                 tk = bet * tk;
```

```
335                 f2 = ((norm (xk + tk*dk-p1))^2 -d1^2)^2 +((norm (xk + ...
                        tk*dk-p2))^2 -d2^2)^2 +                    ((norm (xk ...
                        + tk*dk-p3))^2 -d3^2)^2 +((norm (xk + tk*dk-p4))^2 ...
                        -d4^2)^2   ;
336                 aux =  f + tk * alf * (gf' * dk);
337
338             end
339
340             xk = xk + tk*dk;                %update x and n
341
342             px = xk(1);
343             py = xk(2);
344             n=n+1;
345
346         end
347
348 %from the optimization we get a value from the position q, which will ...
      the the output of the %system
349
350     end
351
352
353
354
355 end
356  myflag2=trig;
357
358  q = [ px;py ];
359
360 end
```

Kalman Filter

```
1
2  function [error, xhatOut, xcorrente ] = EXTKALMAN(q,trigger,vr)
3  %EXTKALMAN Extended Kalman Filter
4
5
6  % Initialization
7  persistent P;
8  persistent xhat;
9  persistent triggerold;
10
11 if isempty(P)
12
13
14 %  xhat = [q; vr];
15
16 xhat=[-50; -30;0;0];
17 P = 1*eye(4);
18 triggerold=0;
19
20 end
21
22 %  Radar update time ∆t is inherited from model workspace
23
24 % 1. Compute Phi, Q, and R
25 Phi =       [1,0,0.1,0;
26              0,1,0,0.1;
27              0,0,0.1,0;
28              0,0,0,0.1];
29
30 Q = 0.1 * diag([0.01 0.01 0.01 0.01]);
31 R =  1* diag([1 1]);
32
33
34 B = [0.1,0;0,0.1;0,0;0,0];
35 M = [ 1,0,0,0;0,1,0,0 ];
36
37
38 % 2. Propagate the covariance matrix:
39 P = Phi*P*Phi' + Q;
40
41 % 3. Propagate the track estimate::
42 xhat = Phi*xhat+ B*vr;
43
44
45 % % 4 b). Compute observation vector y and linearized measurement ...
       matrix M
```

```
 yhat = M * xhat;

% 4 c).  Compute error (Estimation Error)

error = q - yhat;


%so that the program only runs this every 12 seconds
if (trigger == 1 && triggerold == 0 )

% 5. Compute Kalman Gain:
W = P*M'*inv(M*P*M'+ R);

% 6. Update estimate
xhat = xhat + W*error;

% 7. Update Covariance Matrix
P = (eye(4)-W*M)*P*(eye(4)-W*M)' + W*R*W';
end

triggerold = trigger;

xhatOut = [xhat(1);xhat(2)];
xcorrente = [xhat(3);xhat(4)];
```

Path Implementation

```


function [v,w]  = Spath(t)

persistent x;

    if isempty(x)
        x=0;
    end

    w=0;

    if(t<50*pi+600)

        v=1 ;

    else

        v=0;
    end


```

```
23      if(t<100)
24
25          w=0;
26
27      end
28
29
30      if(t ≥ 100 && t < 10*pi + 100)
31
32          w=0.1;
33
34      end
35
36      if(t ≥10*pi + 100 && t < 10*pi+200)
37
38          w=0;
39
40      end
41
42      if(t ≥ 10*pi+200 && t < 20*pi+200)
43
44          w=−0.1;
45
46      end
47
48
49      if(t≥ 20*pi+200 && t < 20*pi+300)
50
51          w=0;
52
53      end
54
55      if(t≥20*pi+300 && t< 30*pi+300)
56
57          w=0.1;
58
59      end
60
61      if(t≥ 30*pi+300 && t<30*pi+400 )
62
63          w=0;
64
65      end
66
67      if(t ≥ 30*pi+400 && t < 40*pi+400)
68
69          w=−0.1;
70
71      end
```

```
72
73      if(t≥ 40*pi+400 && t < 40*pi+500)
74
75          w=0;
76
77      end
78
79      if(t≥40*pi+500 && t< 50*pi+500)
80
81          w=0.1;
82
83      end
84
85      if(t≥ 50*pi+500 && t < 50*pi+600)
86
87          w=0;
88
89      end
90
91
92  end
```

A.3 Some more simulation results

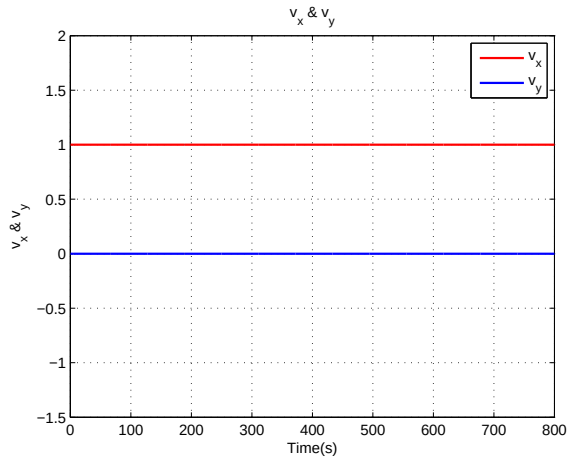

Figure A.4: Velocity main without noise

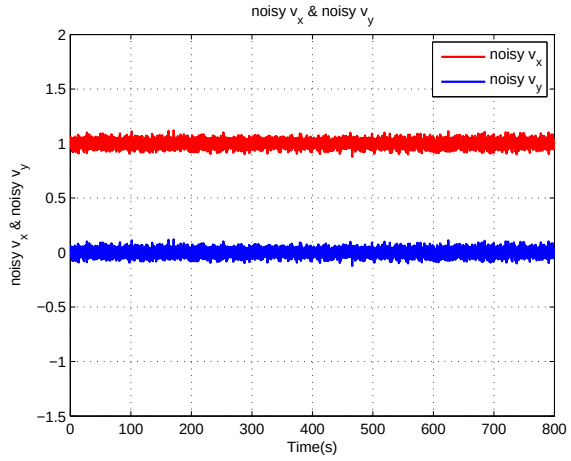

Figure A.5: Velocity with noise

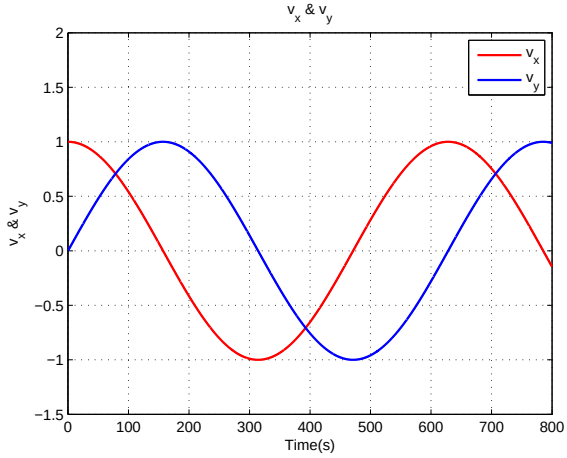

Figure A.6: Velocity without noise

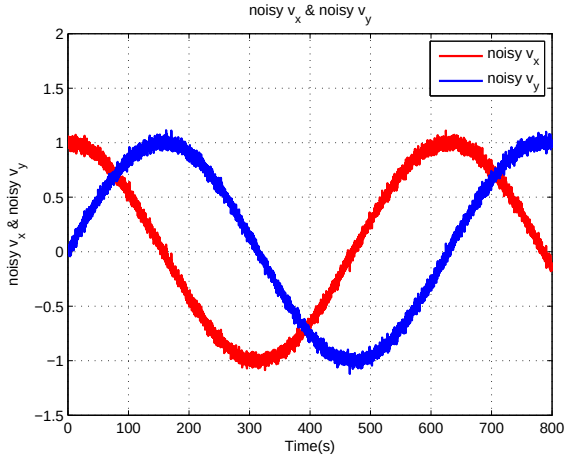

Figure A.7: Velocity with noise

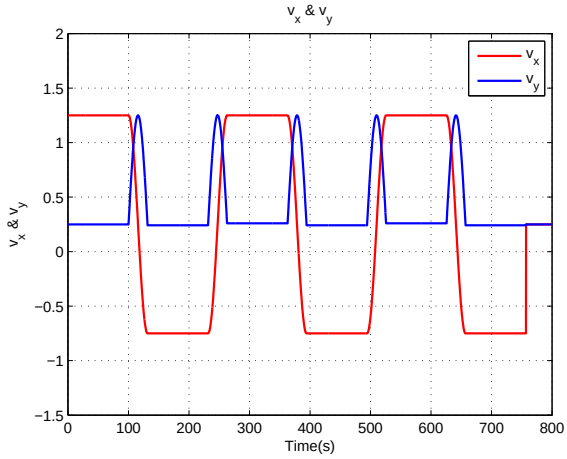

Figure A.8: Velocity without noise

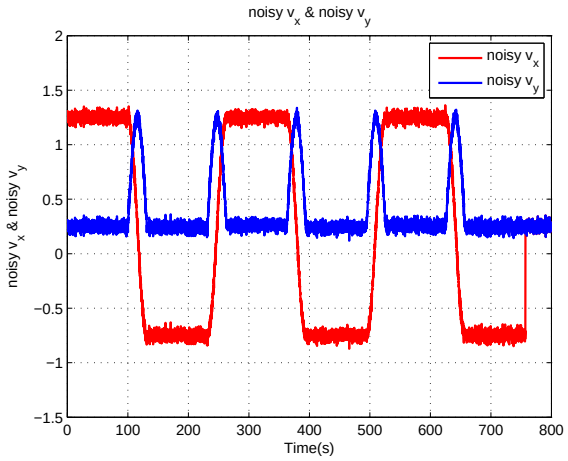

Figure A.9: Velocity with noise

References

[1] A. M. Pascoal and A. P. Aguiar. Eeci graduate school on control supélec, Feb. 21-25 2011. URL: http://goo.gl/Qrb2t.

[2] A. Pedro Saúde, João; Aguiar. Single beacon acoustic navigation for an auv in the presence of unknown ocean currents. URL: http://www.ifac-papersonline.net/Detailed/40375.html.

[3] M. Krishna, J.E. Bares, and E. Mutschler. Tethering system design for dante ii. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 2, pages 1100–1105 vol.2, 1997. doi:10.1109/ROBOT.1997.614283.

[4] G.S. Virk A.K.M. Azad and M.Qi. Robovolc: Remote inspection for volcanoes. URL: http://85.31.145.61/@api/deki/files/2884/=13.pdf.

[5] Brett M. Bethke. Persistent vision-based search and track using multiple UAVs. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge MA, June 2007. URL: http://dspace.mit.edu/handle/1721.1/40392.

[6] O. Schofield, J. Kohut, H. Roarty, S. Glenn, C. Jones, and D. Webb. Enabling discovery based science with webb gliders. In *US/EU-Baltic International Symposium, 2008 IEEE/OES*, pages 1–6, 2008. doi:10.1109/BALTIC.2008.4625486.

[7] A. Aguiary and A. Pascoal. Modeling and control of an autonomous underwater shuttle for the transport of benthic laboratories. In *OCEANS '97. MTS/IEEE Conference Proceedings*, volume 2, pages 888–895 vol.2, 1997. doi:10.1109/OCEANS.1997.624110.

[8] NASA. Doppler effect, 2004. URL: http://www.grc.nasa.gov/WWW/k-12/airplane/doppler.html.

[9] M. Stojanovic. Recent advances in high-speed underwater acoustic communications. *Oceanic Engineering, IEEE Journal of*, 21(2):125–136, 1996. doi:10.1109/48.486787.

[10] A.P. Aguiar and A.M. Pascoal. Coordinated path-following control for nonlinear systems with logic-based communication. In *Decision and Control, 2007 46th IEEE Conference on*, pages 1473–1479, 2007. doi:10.1109/CDC.2007.4434835.

[11] A.P. Aguiar and J.P. Hespanha. Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *Automatic Control, IEEE Transactions on*, 52(8):1362–1379, 2007. doi:10.1109/TAC.2007.902731.

[12] Antonio Pedro Aguiar and J.P. Hespanha. Logic-based switching control for trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. In *American Control Conference, 2004. Proceedings of the 2004*, volume 4, pages 3004–3010 vol.4, 2004.

[13] W.D. Wilson. The speed of sound in sea water, 1960. URL: http://www.akin.ru/spravka_eng/s_i_svel_e.htm.

[14] Ronald K. Jurgen Donald Christiansen, Charles K. Alexander. Standard handbook of electronic engineering, fifth edition, 1960. URL: http://accessengineeringlibrary.com/browse/standard-handbook-of-electronic-engineering-fifth-edition#fullDetails.

[15] F. Thomas and L. Ros. Revisiting trilateration for robot localization. *Robotics, IEEE Transactions on*, 21(1):93–101, 2005. doi:10.1109/TRO.2004.833793.

[16] URL: http://oceansys.fe.up.pt/technology.php.

[17] URL: http://lsts.fe.up.pt/.

[18] URL: http://lsts.fe.up.pt/vehicles/lauv.

[19] URL: http://www.noptilus-fp7.eu/default.asp?node=page&id=1&lng=2.

[20] Cara Elizabeth Grupe LaPointe. Vlbl autonomous underwater vehicle navigation using a single transponder. URL: http://hdl.handle.net/1721.1/35310.

[21] Guoqiang Mao, S. Drake, and B. D O Anderson. Design of an extended kalman filter for uav localization. In *Information, Decision and Control, 2007. IDC '07*, pages 224–229, 2007. doi:10.1109/IDC.2007.374554.

[22] A. Bahr, M.R. Walter, and J.J. Leonard. Consistent cooperative localization. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 3415–3422, 2009. doi:10.1109/ROBOT.2009.5152859.

[23] URL: http://paginas.fe.up.pt/~ee04134/index_ficheiros/Page595.htm.